NetBackup Vault Extension
System Administration Guide
For UNIX Servers
Revision 3.4

F. Bryan Cooper
Brian Blake

Revision October 20, 2000

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

IMPORTANT NOTE TO USERS

While every effort has been made to ensure the accuracy of all information in this document, VERITAS assumes no liability to any party for any loss or damage caused by errors or omissions or by statements of any kind in this document, its updates, supplements, or special editions, whether such errors are omissions or statements resulting from negligence, accident, or any other cause.
VERITAS further assumes no liability arising out of the application or use of any product or system described herein; nor any liability for incidental or consequential damages arising from the use of this document. VERITAS disclaims all warranties regarding the information contained herein, whether expressed, implied or statutory, including implied warranties of merchantability or fitness for a particular purpose. VERITAS makes no representation that the interconnection of products in the manner described herein will not infringe on existing or future patent rights, nor do the descriptions contained herein imply the granting or license to make, use or sell equipment constructed in accordance with this description.

VERITAS reserves the right to make changes without further notice to any products herein to improve reliability, function, or design.

TRADEMARKS

VERITAS, VxVM, VxVA, VxFS, FirstWatch, and the VERITAS logo are registered trademarks of VERITAS Software Corporation.

VERITAS Volume Manager, VERITAS File System, VERITAS Quick I/O, VERITAS NetBackup, VERITAS HSM, and VxSmartSync are trademarks of VERITAS Software Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Co. Ltd.

Other products mentioned in this document are trademarks or registered trademarks of their respective holders.

Table of Contents

New Features

The following new features have been added to NetBackup Vault Extension (also known as bpvault) since the last release:

1. Full support for NetBackup 3.4.

If you are using NetBackup 3.4, you must make a change in your parameter file. The following line should be changed to:

```
nb_version 34
```

If you are running a previous version of NetBackup, please set the parameter as such:

```
NetBackup version            nb_version
2.1 or older                 2
3.0, 3.1, 3.1.1, 3.1.5       3
3.2                          32
3.4                          34
```

The template parameter files (e.g. dup_param.tld_example) now assume you are at least running NetBackup 3.4.

You will also need to update your robot_inventory scripts in ../netbackup/vault/bin. This is because the output of the vmcheckxxx command changed between NBU 3.2 and 3.4. Please use the robot_inventory scripts with the "_34" extension as a template. You may need to copy the template multiple times if you have multiple robots.

2. Support for duplication of images with different source server.

A new feature was added to NetBackup 3.2 with jumbo patch 363 that allows the specification of a source server on the command line of bpduplicate. Vault Extension will now support this feature within the parameter file.

In order to use this new feature, altreadhost, you need to determine which server(s) will be doing the reading of the source backup images. Previously, it was required to use the server responsible for the backup to do the reading during duplication.

The generic usage of this feature is:

```
server <server-where-original-exists> 1 ALTREADHOST <server-to-use-for-reading>
```

where:

```
<server-where-original-exists> is the backup server that performed the backup for
a given image
<server-to-use-for-reading> is the server that you wish to use for reading the
original backup image
```

We will explain two basic scenarios in which to use this new feature. If you require further assistance with this feature, please contact Techincal Support.

The first example is one of the most common that would be used:

```
server <server-to-use-for-duping> <drive-pairs> ALTREADHOST <same-server-name>
...
dstunit <stunit-to-use-for-writing> <drive-pairs>
...
dup_cleanup 1
```

where:

```
<server-to-use-for-duping> is the server which you wish to use for both reading
AND writing the backup image(s)
<drive-pairs> is the number of drive pairs attached to <server-to-use-for-duping>
<same-server-name> is the same as <server-to-use-for-duping>
<stunit-to-use-for-writing> is the name of the storage unit attached to the server
listed in <server-to-use-for-duping>
```

So, for example:

```
drive_pairs 2
server backup1 2 ALTREADHOST backup1
dstunit backup1-dlt 2
class all
schedule none
dup_cleanup 1
<... more parameters listed here ...>
```

In this example, the server named "backup1" has at least four (4) drives attached to it and those drives are in the storage unit named "backup1-dlt". There may be more servers in this configuration that are used to backup data in all of the classes, but we have decided to allow "backup1" to do the duplication for ALL images NO MATTER WHICH SERVER did the original backup.

Normally, we would have to specify each server on a separate line with a "server" parameter. Since we now have the ALTREADHOST option, and we specify the "dup_cleanup" parameter, we are able to duplicate all images from all servers on the same server. The "dup_cleanup" parameter is used to assign images that did not match any server listed in the parameter file to servers that were listed in the parameter file. It is this parameter that allows all of the images from servers other than "backup1" to be read and duplicated on "backup1".

The second example can be used if you wish to duplicate images from a specific server on a different server. It would look like such:

```
server <server-where-original-exists> <drive-pairs> ALTREADHOST <server-to-use-
for-reading>
...
dstunit <stunit-to-use-for-writing> <drive-pairs>
```

where:

```
<server-where-original-exists> is the server which owns the original backup images
(i.e. the server which backed up the data)
<drive-pairs> is the number of drive pairs attached to <server-to-use-for-duping>
<same-server-name> is the same as <server-to-use-for-duping>
<stunit-to-use-for-writing> is the name of the storage unit attached to the server
listed in <server-to-use-for-duping>
```

So, for example:

```
drive_pairs 4
server backup1 2 ALTREADHOST backup2
server backup2 1
server backup3 1 ALTREADHOST backup4
dstunit backup2-dlt 3
dstunit backup4-9840 1
class backup1-all
class backup2-all
class backup3-all
<... more parameters listed here ...>
```

In this example, the server named "backup2" has at least six (6) drives attached to it and those drives are in the storage unit named "backup2-dlt". Any images that belong to the server named "backup1" will be read and writen on server "backup2", and two drive pairs will be allocated for duplicating these images. Any images that belong to the server "backup2" will be read and written on itself, and one drive pair will be allocated for duplicating these images. The server named "backup4" has at least two (2) drives attached to it and those drives are in the storage unit named "backup4-9840". Any images that belong to the server named "backup3" will be read and written on server "backup4", and one drive pair will be allocated for duplicating these images. And, it is assumed that all four servers (backup1, backup2, backup3, backup4) share the same robot.

NOTE: THIS FEATURE SHOULD ONLY BE USED IF THE ALTREADHOST SERVER HAS ACCESS TO THE MEDIA WHERE THE ORIGINAL BACKUP IMAGE WAS STORED. THIS USUALLY MEANS THAT THE SOURCE SERVER AND THE ALTREADHOST ARE SHARING THE SAME ROBOT.

3. Support for multiple robots attached to single server.

This feature allows support for duplicating images where the user wishes to only duplicate images that exist in a particular robot. If a media server controls tape drives in two different robots, then in order to duplicate the images, two different parameter files are required (because of multiple robots). When specifying the classes and schedules to duplicate, it is possible that some images could exist in one robot and some exist in another. NetBackup is not aware of which images exist in which robot, so it is necessary to specify which robotic volume group the images exist in.

The parameters that are required to enable this feature are:

```
multrobots_onserver <toggle>
multrobots_onserver_srcgrp <volume-group>
```

where:

```
<toggle> is either 0 (off) or 1 (on)
<volume-group> is a name of a volume group where the source images exist; the
default value for this parameter is the same as the "robot_group" parameter
```

For example, a server named "backup1" is defined. This server is attached to two (2) tape drives in robot 0 (media in this robot are in volume group 00_000_TLD) and two (2) tape drives in robot 1 (media in this robot are in volume group 00_001_TLD). All classes that are backed up on this server are set to "any available storage unit", so the images could be backed up to media in robot 0 or robot 1, depending on the scheduling of jobs.

If we wish to duplicate all of the images for "backup1", we would setup two parameter files. One parameter file would look like such:

```
drive_pairs 1
server backup1 1
dstunit backup1-robot0-dlt 1
class all
schedule none
robot_group 00_000_TLD
vault_group Offsite
multrobots_onserver 1
<... more parameters listed here ...>
```

Since we have specified "multrobots_onserver" as 1 (turned on the feature), all of the images that belong to the server named "backup1" AND are on media that belong to volume group "00_000_TLD" (we didn't specify "multrobots_onserver_srcgrp", so the default is used, which is the same as "robot_group") will be duplicated to the storage unit named "backup1-robot0-dlt".

The second parameter file would look like:

```
drive_pairs 1
server backup1 1
dstunit backup1-robot1-dlt 1
class all
schedule none
robot_group 00_001_TLD
vault_group Offsite
multrobots_onserver 1
<... more parameters listed here ...>
```

Notice the difference of the "dstunit" and "robot_group" parameters. All other parameters are the same since we are duplicating the set of images that were backed up by "backup1"; the only difference is that we are only going to duplicate those images which reside on media that exist in the volume group 00_001_TLD.

4. Support for ACS API on UNIX master server platforms.

It is now possible to eject media from an ACS robot controlled either by ACSLS (on UNIX) or HSC (on a mainframe) when using a UNIX master server. This functionality will replace the ../vault/bin/eject_tapes script in future versions of Vault Extension.

To enable this functionality, the following parameters need to be defined in the parameter file:

```
robot_type acsla
acs_csi_hostname <server>
acs_nodrives_onmaster <0 or 1>
```

where:

```
<server> is the robot control host (mainframe or ACSLS server)
```

if there are no tape drives defined on the master:

```
acs_nodrives_onmaster 1
```

if there are tape drives defined on the master:

```
acs_nodrives_onmaster 0
```

In addition, it is very important to define the following parameters correctly:

```
acs
acs_lsm
acs_cap_0
acs_cap_1
acs_cap_num
acs_cap_count
```

And, the following parameters can be used to control the timeouts used for interaction with ACSLS:

```
acs_ack_timeout
acs_command_timeout
acs_number_tries
```

Descriptions of these parameters can be found in the Appendix.

5. Reports can be placed into repository.

New functionality has been added to the vault process to allow reports to be saved to a repository directory. The following parameters need to be defined in the parameter file to enable this function:

```
run_report_file <0 or 1>
report_dir <directory-pathname>
```

where:

```
if you wish to generate reports in the repository:
       run_report_file 1
if you do not wish to generate reports in the repository:
       run_report_file 0
<directory-pathname> is the fully qualified pathname to the reports directory; the
usual location is /usr/openv/netbackup/vault/reports
```

If "run_report_file" is set to "1", then reports will appear in the path specified by "report_dir". The file names generated can be modified by editing the script bpvault.reports.

6. Ability to change report header names.

This feature controls the names of the different reports generated during the vault session. The following parameters can be defined in the parameter file to override the defaults (which are listed here):

```
distvlt_header Distribution List for Vault
distdtl_header Detailed Dist. List for Vault
distsum_header Summary Dist List for Vault
origejc_header Original Media Eject List
origdtl_header Detailed Original Media Eject List
picklib_header Picking List for Library
pickvlt_header Picking List for Vault
distlib_header Distribution List for Library
invtvlt_header Inventory List for Vault
fullvlt_header Full Inventory List for Vault
compinv_header Complete Inventory List
recovry_header Recovery Report for Vault
```

If these parameters are not listed in the parameter file, or are commented out (the default for the template parameter files), then the default headers will be used.

7. Support for control of reporting from parameter file.

It is now possible to control the report generation through the parameter file. The following parameters can be defined in the parameter file to enable each feature:

```
run_report_print 1
run_report_mail 1
run_report_file 1
```

By setting any of these parameters to "1" enables the generation of the report to either a printer, email address or file repository. Printer and email commands are still defined in bpvault.env (see LPR, MAILPRG and MAILNAMES).

8. Support for multiple ejects on TL8 robots.

We have devised a workaround for the current problem of ejecting multiple pieces of media into a export door on TL8 robots. The core vmchange utility does not support this feature (due to issues with the extend/retract protocol used on these types of robots).

In order to enable this feature, the parameter file should have the following defined:

```
tl8test_robotpath <device-file>
tl8test_host <hostname>
tl8_eject <vmchange or tl8test>
```

where:

```
<device-file> is the device file used to communicate with the robot controller via
SCSI (e.g. /dev/sg/c2t6l0)
<hostname> is the name of the server controlling the robot; do NOT define this
parameter if the master server is controlling the robot
<vmchange or tl8test> set this parameter to tl8test to use the workaround;
otherwise set this parameter to vmchange for normal vmchange ejects which will
only eject one media at a time
```

This workaround will NOT support a robot controlled by an NT media server. It will also not support a robot controlled by a UNIX media server that reports to an NT master server.

9. Support for 128 drive pairs.

Previous versions of Vault Extension allowed a maximum of 16 drive pairs. This version allows a maximum of 128 drive pairs per robot.

10. Injecting into different volume pools.

The bpinject (bpinject-nt on NT masters) script now supports injects into a volume pool other than NetBackup. when calling "bpinject", the last command line argument should be the pool number in

which you wish to inject new media into. This parameter is optional and will default to pool number 1, the NetBackup pool.

So, an example on UNIX would be:

```
bpinject master1 /dev/sg/c2t6l0 master1 0 5
```

where "5" would be the number of the pool that new media should be injected into

11. Support of "schedule all" parameter.

In addition to the reserved schedule "none", the reserved schedule name "all" is now supported. So, if you wish to vault all of your data over a given period of time (all classes and all schedules), you would list the following in the parameter file:

```
class all
schedule all
```

Therefore, defining “schedule all” is the same as defining “schedule none”.

12. Support for 1024 classes.

Previous versions of Vault Extension allowed a maximum of 256 class definitions in the parameter file. It is now possible to define up to 1024 classes in the parameter file.

13. Moved reports to separate script.

Previous versions of Vault Extension required the use of the robot eject script (bpvault.tld.eject, bpvault.acs, bpvault.tl8.eject, bpvault.tlh.eject and bpvault.tlm.eject) to generate the reports.

All report generation is now located in a new script, bpvault.reports. Each of the robot eject scripts will call this script to generate reports. If you wish to modify the way in which reports are generated, this file can be modified (e.g. comment out reports that your site does not use).

14. Commented parameter file templates.

Thanks to Rob Worman of Collective Technologies, we have commented the template parameter files. All parameters are followed by brief comments that define their syntax and purpose.

Introduction

This document provides operational procedures for running the bpvault system. It is divided into the following sections:

1. **System Administration Design** - shows the basic installation, configuration, and troubleshooting responsibilities for Storage Administrator to use bpvault successfully.
2. **System Administration Procedures** - shows detailed information needed by Storage Administrator to follow.
3. **Sample Configuration and Log files**- Shows parameter files and output log files.

We have also provided a **Functional Design** document that illustrates the technical design and an **Operations Guide** that provides day-to-day procedural information.

Background

Bpvault was developed as a utility program for use with NetBackup. Its purpose is to assist in disaster recovery by creating duplicate copies of backup tapes. If backup tapes are destroyed at a primary data center location, duplicates for some of these backups are still available at an offsite location. Bpvault creates the duplicates, ejects them from any robotic storage units, and assigns offsite vault slot numbers. Bpvault keeps track of the duplicates and requests these tapes to be returned from the offsite location after a specified period of time, e.g. when the backup is no longer needed. Bpvault also has the ability to bypass the duplication process and eject the original media. This optional mode is useful for sites that have large amounts of data and a limited window in which to duplicate the backup images.

This System Administration Guide documents how to install bpvault code, to understand the various steps used to create duplicates and to track them in the offsite vault, and how to manually restore images using the duplicated images. It also shows how to manually run the duplication process in case of interruption of the process, and some troubleshooting guidelines.

Notes

This manual uses “shell script like” notations for certain “variables”; most notably are the $VAULT and $DUPID. These notations are defined as follows:

$VAULT – the name of the vault being used for a given duplication session

$DUPID – a unique number (or ID) referring to a duplication session

Design

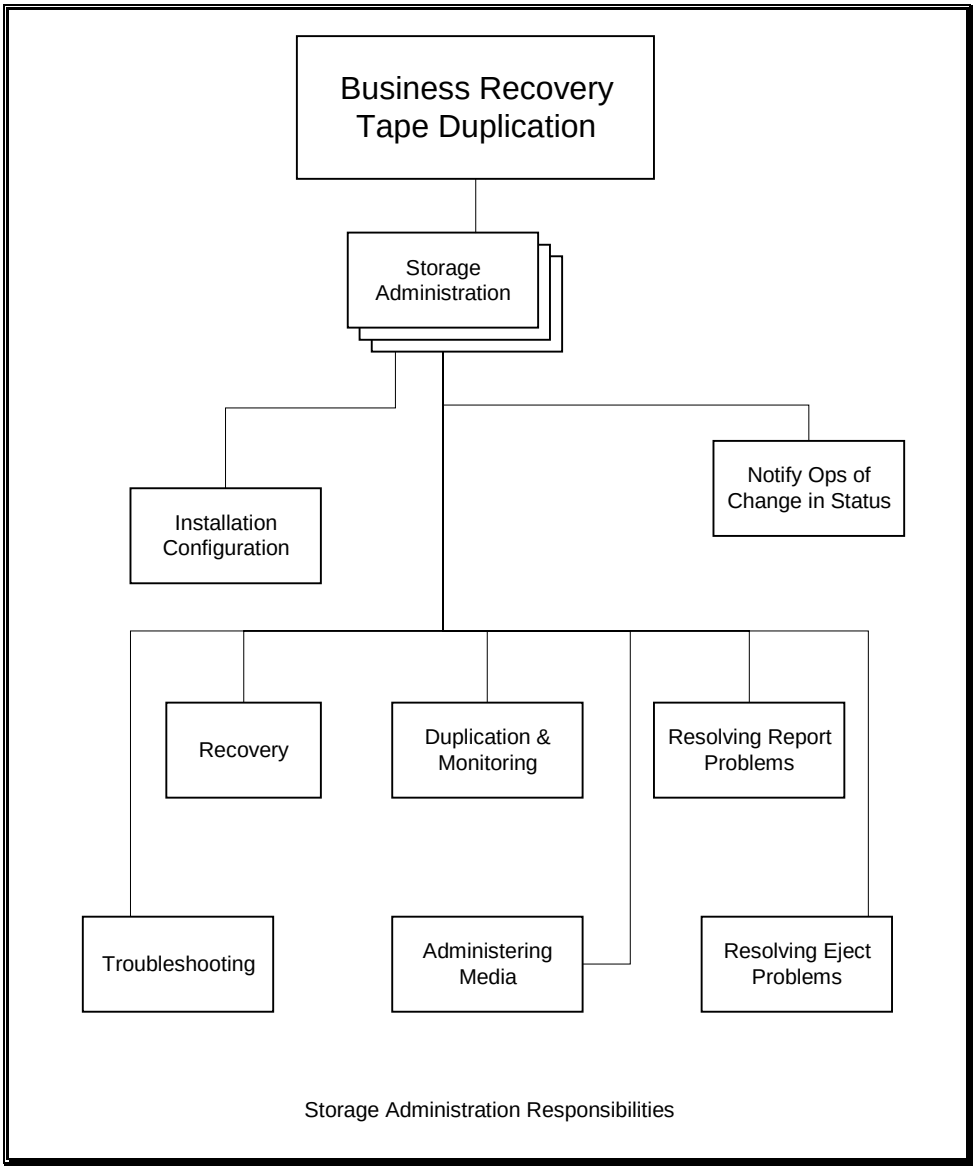
The overall design of bpvault was created to simplify the job for both storage administration and systems operations. Both Storage Administration and Operations responsibilities are summarized here to help Operations understand how to gain necessary support. The separate Operation's Guide provides detailed instructions for Operational Staff. Detailed Storage Administration instructions are provided in the next sections. The following table highlights the different tasks and assigns responsibilities:

Staff Responsibilities
Storage Administration
1. Installation and Configuration of bpvault
2. Running and Monitoring daily duplications to ensure they complete.
3. Administering tape media to ensure sufficient media available for each day's duplicates.
4. Resolving conflicts between printed reports and offsite vendor status.
5. Resolving issues about tapes improperly ejected.
6. Manually recovering duplicated media.
7. Troubleshooting

Support

Please contact VERITAS Customer Support (1-800-342-0652) regarding problems with Vault Extension.

The following diagram shows the overall Operational Design from these responsibilities:



Storage Administration Responsibilities

Procedures – Installation and Configuration

This section outlines the steps required to install and configure bpvault.

Software Installation

Bpvault consists of a primary program, the bpvault binary, several small shell menu scripts, “bpvault.all”, “bpvault.tld”, “bpvault.acs”, “bpvault.menu”, “bpvault.opsmenu”, and a parameter file, e.g. “dup_param”. Bpvault files will be kept under a subdirectory /usr/openv/netbackup/vault. Secondary scripts such as “robot_inventory” and “eject_tapes” are stored in /usr/openv/netbackup/vault/bin. These scripts may need to be customized for specific robots (see “Robot Customization”).

Obtain the bpvault archive and copy it to the directory /tmp on your NetBackup master server. Unpacking (using crypt, uncompress and tar) the archive will load all files into the directory /usr/openv/netbackup/vault.

```
# cp bpvault-dist.tar.Z.y /tmp
# cd /usr/openv/netbackup
# cat /tmp/bpvault-dist.tar.Z.y | crypt | uncompress | tar xvf -
```

When you are prompted with “Enter Key:”, type in the password provided to you by VERITAS Professional Services.

Next, change the current directory to /usr/openv/netbackup/vault/production. Create a symbolic link that links bpvault.<platform> to bpvault, where <platform> is the platform of your NetBackup master server (e.g. solaris, hp800, rs6000, etc.).

```
# cd /usr/openv/netbackup/vault/production

For Solaris 2.X:
# ln –s bpvault.solaris bpvault

For HP-UX on HP9000-800:
# ln –s bpvault.hp800 bpvault

For AIX on RS6000:
# ln –s bpvault.rs6000 bpvault

For NCR:
# ln –s bpvault.ncr bpvault

For SGI:
# ln –s bpvault.sgi bpvault

For DEC:
# ln –s bpvault.dec bpvault

For Sequent:
# ln –s bpvault.sequent bpvault
```

Normal NetBackup software should already be in place and running on the system prior to installing bpvault.

NetBackup Configuration

The first step to setting up bpvault is to configure appropriate NetBackup attributes for use by bpvault and to identify which NetBackup classes you wish to vault.

1. **Volume Pools**: If you wish to use bpvault to duplicate backup images, setup an appropriate pool for use by the duplication system. You can choose any name for the pool, e.g. “Duplicates”. If you wish to use bpvault to eject your original backup media, then you should

identify ALL of the pools that are being used for backups. You will specify the appropriate volume pools for your original media within the parameter file (see parameter “pool” below). You must also specify a volume pool to be used for the NetBackup database backups (see parameter “dbpool” below).

NOTE: Multiple pools (per vault) are only supported for original media ejects.

NOTE: The “dbpool” parameter should be DIFFERENT from any pool(s) listed in the “pool” parameter.

NOTE: If the pool used for duplication of NetBackup images is empty and a scratch pool is defined (see Media Manager System Administration Guide for definition of scratch pools), NetBackup will automatically move media from the scratch pool into the duplication pool. However, this will NOT happen for media in the “dbpool”; the media in this pool must be monitored for use. A future version of Vault Extension will allow for the automatic move of media from the scratch pool into the “dbpool”.

2. **Volume Group**: Place the main set of tapes into a volume group that is robotically controlled, e.g. “robot_grp”. You can add a few tapes to a non-robotic volume group which will become the vault group, e.g. “vault_grp”. If the non-robotic volume group does not exist, it will be created automatically when tapes are moved offsite at the end of the bpvault session. Most users will use the automatically generated volume group that corresponds to the media within the robot, e.g. “00_000_TLD”.
3. **NetBackup Classes**: Initially choose a couple of classes to test the bpvault program. It is simple to add more classes later. Use the command “bpcllist” (located in /usr/openv/netbackup/bin/admincmd) to get a full list of classes. Note that the retention period for the class is the same for the duplicate copy, so you may wish to start with classes having a short retention period.
4. **NetBackup Servers**: Bpvault is designed to duplicate on the same server/storage unit that created the original backup image. However, it is possible to duplicate over the network if you list a slave storage unit in the parameter file. Due to bandwidth considerations, double check the “dstunit” parameter for accuracy. You need to know how many drive pairs are available on each server. Use the command “bpstulist -U” (located in /usr/openv/netbackup/bin/admincmd) for a printout of the existing storage units.

Parameter File Configuration

Overview

The basic bpvault configuration requires setting up a parameter file to show which servers to use, how many drive pairs per server, which classes to duplicate (or eject), the name of the duplicate pool, the name of the vault and the location for the vault working subdirectories. Bpvault can have several vaults running on the same server simultaneously as long as they have different vault names and different pools. The parameter files are usually located in the directory “/usr/openv/netbackup/vault/production”. Please refer to the Appendix for discussion of each item in the parameter file.

There are several example parameter files located in the “/usr/openv/netbackup/vault/production” directory. The extension of the file name describes the configuration that it represents. For example, “dup_param.tld_example” shows an example of a vault configuration using a TLD robot. It is recommended to copy one of these files that corresponds to your configuration. The recommended name for the parameter file is “dup_param”, but the file name is arbitrary since it is specified when executing bpvault. If you have multiple vaults, you may wish to append the vault name to the end of the parameter file (as mentioned below), e.g. “dup_param_V1”, “dup_param_V2”, etc.

Vault Naming

When modifying the parameter file, you will need to select a name for the vault. The standard naming convention is V1, V2, V3, etc., however the name can be anything. The vault name is restricted to five characters or less due to a Media Manager limitation with the description field. After you select a name for the vault, specify that name in the parameter file (parameters “vault” and “bpvault_dir”). The directories for these vaults will be automatically created in /usr/openv/netbackup/vault the first time you run the vault program.

For sites that have multiple robots within a NetBackup cluster (e.g. one master, multiple slaves), multiple parameter files may be needed. This is because bpvault requires that each unique robot be defined as a separate vault. This parameter file can be specified on the command line to the bpvault program and most bpvault scripts. So, for each unique robot, a separate vault will need to be created along with a separate parameter file. Usually, it is recommended to use the vault name within the parameter file to help determine which parameter file is used with which vault. For example, if the name of the vault was “CH_V1” as above, then a recommended name for the parameter file would be “dup_param_CH_V1”.

Vault Duplicates or Originals

Bpvault allows you to vault either the duplicate or original media. There are circumstances that may require a site to send the original media offsite instead of duplicating the backup images and sending the duplicate media offsite. One of the main reasons for this is related to the time involved with duplication. If the window allocated for duplication is smaller than the actual time it takes to do the duplication, then it may require the site to send the original media offsite. If you wish to vault duplicates for some classes and vault originals for other classes, you will need a separate parameter file for each process.

To vault originals, the following steps should be taken:

1. The parameter “vault_type” should be defined as “original”, as opposed to the default value of “normal”. The bpvault.all script has logic built in that will use this parameter and skip the duplication process. The inventory process will still be executed since a list of images and media are required prior to generating reports and ejecting tapes.
2. The parameter “vault_group” must be defined appropriately. The filter process will be executed after the inventory process is complete, which will filter out all images that are contained on media that has already been sent to the vault (i.e. media that belongs to the “vault_group” volume group).

   NOTE: THE VALUE FOR “vault_group” MUST BE LESS THAN (or equal to) TEN (10) CHARACTERS IN LENGTH.
3. The “pool” parameter should be defined as the pool of tapes on which the data resides. Since we may be vaulting a number of different classes, multiple “pool” parameters are allowed. It should be noted that if a number of images resides on a given media and these images consist of multiple classes, bpvault will still send this media offsite if only one class (listed in the parameter file) matches. That is, if media ABC123 has six images on it which were backed up from class “classone” and “classtwo”, and “classone” is listed in the parameter file, then media ID ABC123 would be sent offsite. It is recommended that such classes be directed to a different volume pool when defining the class. This way, only classes that are sending data to this separate pool will be ejected from the library and sent offsite. The pool that is selected for these particular backups should be defined using the “pool” parameter. Multiple “pool” parameters can be defined, one per line. For example, if your configuration has backups being directed to volume pools named “NetBackup”, “RDBMS” and “PC”, your parameter file should look like this:

```
...
pool NetBackup
pool RDBMS
```

```
pool PC
...
```

4. The “drive_pairs”, “server” and “dstunit” parameters should be set appropriately, but will not be used. Even though these parameters are not used in processing of original images, it is recommended to set these parameters appropriately for the vault configuration. Most users will set the “drive_pairs” to 1, the “server” will be defined as one of the NetBackup servers in the configuration and the number of pairs on this server will be set to 1, and the “dstunit” parameter will be defined as the storage unit corresponding to the “server” listing above and the number of pairs in this storage unit will be defined as 1.
5. All other parameters should be set according to the general recommendations below.

To vault duplicates, the following steps should be taken:

1. The parameter “vault_type” should be defined as “normal”.
2. The “drive_pairs”, “server” and “dstunit” parameters should be defined appropriately.

The “drive_pairs” parameter should be set to the TOTAL number of drive pairs within this parameter files. This should include the drive pairs on all servers listed in the “server” parameters.

Each “server” that was used for backups should be listed in this parameter file. See the note in General Recommendations below about defining multiple names for the server, e.g. for servers with more than one network interface. The corresponding storage unit should be listed as a “dstunit” parameter. In most configurations, the “server” and “dstunit” parameters are a one-to-one mapping.

When defining the “server” and “dstunit” parameters in the parameter file, it should be noted that the order of these parameters is important. The first “server” entry will match up with the first “dstunit” entry; the second “server” entry will match up with the second “dstunit” entry and so forth. The “server” entry is used to determine which server the backup image was originally written on, and the “dstunit” entry identifies where the backup image should be written to.

If a parameter file was defined as such:

```
drive_pairs 3
...
server ServerA 1
server ServerB 2
...
dstunit ServerA-DLT 1
dstunit ServerB-HCART 2
...
```

The backup images that were written on ServerA will be collected in a file “bpvault.dup.x” (see “Procedures – Duplication”) and the images written on ServerB will be collected in a file “bpvault.dup.y”. When the actual duplication process is started, bpvault will call the bpduplicate interface with the destination storage unit “ServerA-DLT” for the images within file “bpvault.dup.x”, and “ServerB-HCART” for the images within the file “bpvault.dup.y”.

3. Determine if duplicate images will be multiplexed or not.

With NetBackup 3.2 and above, the duplication process can either retain the multiplexed format of multiplexed images or the images can be de-multiplexed. Previous versions of NetBackup only allowed for the de-multiplexing of images.

To enable the multiplexed duplication feature, the following parameter should be defined in the parameter file:

```
mpxdup 1
```

By default, this parameter is not defined in the parameter file. Therefore, de-multiplexing of images is the default behavior. If you wish bpvault to de-multiplex the images during duplication, you can either remove the parameter from the parameter file, or define it as such:

```
mpxdup 0
```

The multiplexed duplication process will occur for all multiplexed images that are selected for duplication during a given bpvault session.

NOTE: If this feature is enabled, make sure that ALL destination storage units (dstunit in parameter file) have multiplexing enabled. Otherwise, an error (unimplemented feature) will be generated in the output file.

4. Determine if duplicate images will retain the same expiration date as the original images.

With NetBackup 3.2 and above, each image copy has a separate expiration date. When the duplicate is created, this date will be the same as the original. However, if the parameters below are defined in the parameter file, the NetBackup catalog will be updated and the duplicate copies will have a different expiration date than the original.

The following parameter must be defined in the parameter file to enable this feature:

```
changeexpdate 1
```

Next, the following parameters should be set accordingly:

```
retention_length 0 xxx
retention_length 1 xxx
retention_length 2 xxx
retention_length 3 xxx
retention_length 4 xxx
retention_length 5 xxx
retention_length 6 xxx
retention_length 7 xxx
retention_length 8 xxx
retention_length 9 xxx
```

The “xxx” shown above is the number of days that will be **ADDED** to the current expiration date of the NetBackup image. This number can be **POSITIVE** or **NEGATIVE**. The default value of “xxx” is zero (0). If a particular retention level is not defined in the parameter file, the number of days will be assumed to be the default, zero (0).

After the completion of the duplication process, bpvault will change the expiration dates of any images that were successfully copied, based on their retention level.

For example, you have a number of backup images set to retention level 1, and retention level 1 is defined as two (2) weeks (or 14 days). If you wish to set the expiration date of the duplicate copies to 30 days, then you would define the following in the parameter file:

```
retention_length 1 16
```

This is because the current expiration date of the image is 14 days from today, and we need to add 16 days to this, which will cause the expiration date to be set to 30 days from today.

If you wish to set the expiration date of a set of images (within a retention level) to infinity, then define the following in the parameter file:

```
retention_length X 99999
```

where “X” represents the retention level of the images you wish to modify.

NOTE: There is a known issue in which setting the retention_length to a NEGATIVE number (where duplicate images would expire BEFORE original images) will not correctly update the media expiration date, which is the date listed on the reports. The problem is being investigated by Vault Extension and NetBackup engineering to work on a fix.

The current workaround is to run “bpexpdate –deassignempty” to make sure that the media will be expired properly and be returned from the vault.

General Recommendations:

1. If you wish to have bpvault.all (the main script) eject media automatically at the completion of a session, then set the “robot_eject” parameter to “auto”. Otherwise, if you wish to use bpvault.opsmenu (see Operations Guide for more details), then set “robot_eject” to “manual”.

   If you believe you will be ejecting more media than the capacity of the robot’s service port, then it is recommended to set “robot_eject” to “manual” and then use bpvault.opsmenu to eject media and print reports. For robots that are controlled by Media Manager (e.g. TLD, TL8), the media must be ejected prior to generating reports. This is due to the use of the volume group to track offsite media.

2. The “server” parameter can be defined with multiple host names. This may be necessary for servers which have multiple network interfaces.

   For example, our master server has two interfaces named "nbumaster1" and "nbumaster1-atm", and one of our slaves has three interfaces named "nbuslave2", "nbuslave2-fddi", "nbuslave2-net13". Our master server has 4 drives (or two pairs) dedicated for duplication and our slave server has 2 drives (or 1 pair) dedicated for duplication. The resulting parameter file should look like:

   ```
   ...
   server nbumaster1 2 nbumaster1-atm
   server nbuslave2 1 nbuslave2-fddi nbuslave2-net13
   ...
   ```

   PLEASE NOTE: MAKE SURE YOU LIST A GIVEN SERVER NAME ONLY ***ONCE* IN THE PARAMETER FILE. USING THE SAME SERVER NAME ON DIFFERENT "server" LINES MAY CAUSE PROBLEMS.**

   As mentioned in the Appendix, the syntax for the “server” parameter is:

   ```
   server <hostname> <number-pairs> [<alternate-name> ... <alternate-name>]
   ```

   where:

   ```
   <hostname> is the hostname of the server
   <number-pairs> is the number of drive pairs to be used for duplication
   <alternate-name> is the other interface names of this server, one for each network
   interface
   ```

3. The “class” parameter should be defined as the NetBackup class(es) that you wish to vault. Each class should be listed on a separate line.

   If you wish to vault all classes, then you can use the following definition in the parameter file:

   ```
   class all
   ```

   NOTE: This usage assumes that your NetBackup configuration does not contain a class named “all”.

4. The “schedule” parameter should be defined as the NetBackup schedule(s) that you wish to vault. Each schedule should be listed on a separate line.

   If you do NOT wish to specify schedules, then you can use the following definition in the parameter file:

   ```
   schedule none
   ```

   NOTE: This usage assumes that your NetBackup configuration does not contain a schedule named “none” within any classes.

   When this option is used all schedules for a given class will be selected when specifying that class in the parameter file during preview mode.

5. The “duplicate_days”, “duplicate_hours”, “days_startfrom” and “hours_startfrom” parameters should be defined as the range of days and hours to search for backup images within the NetBackup catalog.

   The “duplicate_days” parameter defines the furthest number of days to search in the past FROM TODAY. “duplicate_hours” is a number of hours ADDED TO "duplicate_days". These two parameters set the END date of which to search the NetBackup catalog.

   The “days_startfrom” parameter defines from which day to start the search. “hours_startfrom” is the number of hours ADDED TO "days_startfrom". These two parameters set the START date of which to search the NetBackup catalog. Seven (7) days is a typical window that ensures retries for duplicating images missed or which may have had media missing or in use errors.

   All parameters are relative to the current time.

   For example, if it is currently 1/30/99 at 07:00:00, and you wish to vault (either duplicate or originals) images from yesterday night (1/29/99) after 8pm (20:00:00) and only look for images until this morning at 4am (04:00:00), then you would set the following parameters:

   ```
   duplicate_days 0
   duplicate_hours 11
   days_startfrom 0
   hours_startfrom 3
   ```

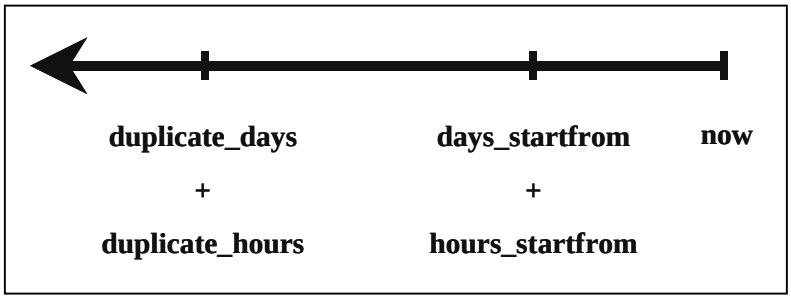
   This would make the search dates start at 01/29/99 20:00:00 and end at 01/30/99 04:00:00.

   Therefore, the *_startfrom parameters should be LESS THAN the duplicate_* parameters.

   NOTE: The “duplicate_hours” and “hours_startfrom” parameters are provided for increased granularity; it is not mandatory that they be used. If these parameters are NOT defined in the parameter file, their values will default to zero (0).

   It is recommended to set up these parameters in the parameter file, and then test them using "bpvault.menu" to create a new duplication session (option 5) and run the preview mode (option 6). Then, look at the "dup.preview.out" file in the appropriate session working directory (e.g. /usr/openv/netbackup/vault/V1/DUP87) to determine if the search parameters are what you expected.

The following diagram illustrates the use of these parameters:



6. The parameters “dbpool”, “num_dbdups”, “days_holddbtape”, “dbbackuphost” and “dbpath” should be defined if you wish to use Vault Extension to send a copy of the NetBackup and Media Manager databases offsite.

   For a detailed discussion of these parameters, please see the section entitled “Procedures – Vaulting NetBackup Databases”.

7. If the master server does not contain the Volume Database (set from tpconfig/xdevadm and known as “Volume Database Host” for a particular robotic device) for a particular robot, you MUST define the following parameter:

   ```
   mmhost <hostname>
   ```

   where:

   ```
   <hostname> is the host on which the volume database resides for this robot.
   ```

8. As noted above, the value for the “vault_group” parameter MUST BE less than or equal to TEN (10) characters in length. If the value is more than ten characters, the media will not be ejected properly and the Media Manager database will assume these media still reside within the robot.

Robot Configuration

Depending on the type of robotics you are using, please use the following recommendations:

All Robot Types

1. If the master server is configured with multiple robots (either controlled by the master or by slave servers), then bpvault must be configured properly. As mentioned above, multiple parameter files and multiple vault directories will be needed. Each unique robot must be defined as a separate “vault”. That is, all of parameters within a given parameter file must be specific to a unique robot such as “robot_num”, “robot_group”, and so forth. Each vault should be setup as mentioned in the installation section of this manual. Be sure to create the vault directory in /usr/openv/netbackup/vault (e.g. V1, V12, etc.).

   Offsite slot IDs will be unique only within a given vault. For example, when ejecting tapes from robot 1 (e.g. V1 in bpvault) the slot IDs will normally be assigned starting at slot 1. When ejecting tapes from robot 2 (e.g. V2 in bpvault), the same slot IDs could be generated even though both robots are within the same NetBackup master-slave configuration. Normally, customers will ask their offsite vendor to setup a separate vault for each group of tapes. This also guarantees that tapes that were ejected from one silo will be injected back into the same silo.

Separate volume pools and volume groups will be needed for each unique robot, or vault. The reason for this requirement is that if multiple vaults are running at the same time the assignment of offsite slot numbers could also happen at the same time. If all of the tapes used for duplication are in the same volume pool, duplicate slot numbers could be assigned to different media. To prevent this from happening, make sure each vault uses a different volume pool and volume group for the duplicated media (see the parameters “pool” and “robot_group” below).

The parameter “robot_inventory” should be defined as a different inventory script in each parameter file. Each robot inventory script should be used to inventory a unique robot. This “helper” script is described below.

2. The robotic inventory script contains commands that need to be executed for each unique robot. Therefore, if a robot is added to the site configuration, it is required that the appropriate additions be made to the “robot_inventory” script. For examples of these scripts, please see the directory /usr/openv/netbackup/vault/bin, and examine the files “robot_inventory.acs” (for ACSLS robots), “robot_inventory.tld” (for TLD robots) and “robot_inventory.tld_mult” (multiple TLD robots). Once you determine which “robot_inventory” script will work for you, copy the appropriate script to “robot_inventory” (e.g. if we were using an ACS controlled robot):

```
# cd /usr/openv/netbackup/vault/bin
# cp robot_inventory.acs robot_inventory
```

NOTE: It is imperative that once you have copied the appropriate script that it is modified to work with your site configuration. This means that the hostnames and robot numbers (where applicable) are modified within the script. It is also recommended that you execute this script from the command line to verify that it is operating correctly. The output of the “robot_inventory” script should be a list of media IDs, with one media ID per line.

This script is used to verify the media that are in the robot. When media are brought back from the vault, this inventory is executed to compare the contents of the robot with the contents of the Media Manager database. If bpvault finds media within the robot (for the pools listed in the parameter file) that are listed as “non-robotic” in the Media Manager database, it will update the Media Manager database for these media. If these media are expired tapes (those that have returned from the vault), it will also reset their description field (offsite slot number, date requested, etc.).

ACS Robotics

For ACSLS robots, you must modify the file “/usr/openv/netbackup/vault/bin/eject_tapes” to include the correct hostname for the ACSLS server (see the “REMSH” command) and the correct CAP door that is queried (see the “TEST” command, “grep” statement). The lines mentioned are listed below:

```
# Remote shell command
REMSH="rsh -l acsss acsls-server"

# Check status of CAP and verify number of Slots in ATL
TEST=`echo $CAP | $REMSH /export/home/ACSSS/bin/cmd_proc -l -q 2> /dev/null | grep
"0, 0,0" | awk '{print $7}'`
```

It is also required that the ACSLS server includes an rhosts entry for the NetBackup master server. For example, if your master server is called “nbserver” and your ACSLS server is called “acsserver”, you should add the following line to the file “.rhosts” in the home directory for user “acsss” on the server “acsserver”:

```
nbserver root
```

This will allow the NetBackup master server to login to the ACSLS server as the user “acsss” and execute the “cmd_proc” utility to eject tapes. This will also allow the NetBackup master server to access the “volrpt” command which is used in the “robot_inventory” script.

If your site has multiple ACS robots, a separate eject_tapes script should be defined for each ACS robot. You can copy the standard eject_tapes script to a new file (e.g. eject_tapes.robot2) and then define the “eject_command” parameter within the parameter file for that robot.

Certain parameters need to be defined properly in the parameter file. These include the following:

```
robot_type
acs
acs_lsm
acs_cap_0
acs_cap_1
acs_cap_count
acs_cap_num
```

Please refer to the Appendix for more information about these parameters.

Media Manager Controlled Robotics (TLD, TL8)

The tape ejection script for TLD and TL8 robots will be generated after the completion of a duplication session, and resides in the directory corresponding to the vault name and session ID (e.g. /usr/openv/netbackup/vault/V1/DUP27), also known as the session working directory.

If you have multiple TLD or TL8 robots, the robot_inventory.tl* scripts can be used as templates for creating the robot_inventory script that is unique to each robot. Once you copy this template to a new file (e.g. robot_inventory.robot1, robot_inventory.tl8-1, etc.), then define the “robot_inventory” parameter within the appropriate parameter file.

For sites with TLD controlled robots, the script /usr/openv/netbackup/vault/production/bpinject should be modified. This script contains operating system specific commands (such as remote shell, grep, etc.). Please edit this script prior to using it to be sure that the environment variables are configured correctly (especially RSH and GREP). For a further discussion of this script, please see the section entitled “Procedures – Miscellaneous”.

The standard eject command that is used to eject tape from TLD and TL8 robots (vmchange) will eject tapes from the library into the mail (export) slots. For smaller robots that have only 1 mail slot, the robot may re-insert the tape from the mail slot if human intervention does not occur within a certain amount of time.

A timeout parameter can be modified to allow for an operator to remove the media from the mailslot so that the robot does not automatically re-insert the tape. This parameter is passed to the "vmchange" command which is sent to the robot to allow for the extra time.

The following parameter should be added to your parameter file to use the option:

```
vmchange_timeout 5
```

The default value for this parameter is 5 (seconds), so this parameter does not need to be defined if you wish to use the default. If you are having this type of problem with your robot, try increasing this parameter to, for example, 30 (seconds).

Certain parameters need to be defined properly in the parameter file. These include the following:

```
robot_type
tld_count (for TLD robots)
tl8_count (for TL8 robots)
```

Please refer to the Appendix for more information about these parameters.

TLH Robotics

The tape ejection script for TLH robots will be generated after the completion of a duplication session, and resides in the directory corresponding to the vault name and session ID (e.g. /usr/openv/netbackup/vault/V1/DUP27), also known as the session working directory.

If you have multiple TLH robots, the robot_inventory.tlh scripts can be used as templates for creating the robot_inventory script that is unique to each robot. Once you copy this template to a new file (e.g. robot_inventory.robot1, robot_inventory.robot2, etc.), then define the “robot_inventory” parameter within the appropriate parameter file.

It is imperative to define the robotic "host"; this is the host that has the device file (usually /dev/lmcp0) present. In most configurations, this host is a UNIX machine running Solaris or AIX. To determine which host is controlling the TLH robot, please use the Media Manager configuration utility "tpconfig". If you run "tpconfig -d", a brief configuration summary screen will be displayed to show you which host is controlling the robot.

Certain parameters need to be defined properly in the parameter file. These include the following:

```
robot_type
mtlib_host
mtlib_command
mtlib_device
mtlib_bulk
remote_command
tlh_count
```

Please refer to the Appendix for more information about these parameters.

TLM Robotics

The tape ejection script for TLM robots will be generated after the completion of a duplication session, and resides in the directory corresponding to the vault name and session ID (e.g. /usr/openv/netbackup/vault/V1/DUP27), also known as the session working directory.

If you have multiple TLM robots, the robot_inventory.tlm scripts can be used as templates for creating the robot_inventory script that is unique to each robot. Once you copy this template to a new file (e.g. robot_inventory.robot1, robot_inventory.robot2, etc.), then define the “robot_inventory” parameter within the appropriate parameter file.

It is imperative to define the hosts that are responsible for communicating with the DAS software. The “dasadmin_client” is the host where the “dasadmin” command is located and where it will be called from; this is usually the master server. The “dasadmin_server” is the OS/2 machine that is physically connected to the library.

Certain parameters need to be defined properly in the parameter file. These include the following:

```
robot_type
dasadmin_command
dasadmin_insert
dasadmin_eject
dasadmin_eject_complete
media_type
dasadmin_client
dasadmin_server
tlm_count
```

Please refer to the Appendix for more information about these parameters.

To support any other robotic configurations, these scripts will need customization available from the VERITAS Professional Services Organization.

Customization

Most of the bpvault shell scripts can be used without any modifications. However, one file in particular should be modified for site specific configuration. This file located in "/usr/openv/netbackup/vault/production/bpvault.env" contains a number of shell environment variables that are used during the execution of bpvault. The variables are used for various purposes, such as network printer names, email addresses, etc. Please review this file and make the appropriate changes for your environment. All lines are commented so it should be self-explanatory as to which variables should be modified.

Another script which you may wish to modify is the report generation script, which is "bpvault.reports". The modifications within this script should only involve removing (or commenting out) the generation of certain reports. All reports that are generated have the same format within this script:

```
#
# BPVAULT- report picking_library
# Picking library report: pick list for tapes going offsite
#
  $bpvault -dup_id $DUPID -function report -command picking_library –param
$dup_param -file $file -output $output > $TEMPFILE
  $MAIL -s "Bpvault Eject Report Session $DUPID" $MAILNAMES < $TEMPFILE
  $LPR $TEMPFILE
```

Each report is generated using the bpvault command line interface and redirected to a temporary file. This file is then sent to the mail program (specified in bpvault.env) and then sent to the printer (using the parameter specified in bpvault.env). If you do not wish to generate a particular report, simply place a comment (pound sign, or #) at the beginning of each of the three lines. Then, if you change your mind later, you can simply remove the comments to generate the report.

Please see the comments within these scripts for more details.

Power Tip

Execution and Scheduling

The main bpvault script should be executed from either the command line or from a scheduler (e.g. cron; see below). This script requires a command line argument, which is the parameter file (e.g. dup_param). For example, if you would like to initiate a new session of bpvault using the parameter file dup_param_V1, use the following syntax:

```
# cd /usr/openv/netbackup/vault/production
# ./bpvault.all dup_param_V1
```

This script will accomplish the following tasks:

1. Build a list of backup images to process based on the class(es), schedule(s) and date range.
2. Process these images either by duplicating them to a new set of media, or by building a list of media that the images reside on.
3. Inventory the NetBackup and Media Manager databases to determine which media should be vaulted.
4. If requested, perform a backup of the NetBackup and Media Manager databases to be sent offsite.
5. Eject media from the library to be sent offsite.
6. Generate reports describing the media to be sent offsite, media to be returned from the vault and miscellaneous inventory reports.

Scheduling of Bpvault

Currently, NetBackup does not support automatic scheduling of bpvault processes. Most sites are using a scheduler (such as cron, or a third-party scheduler) to initiate a bpvault session. One example of how to schedule this process with cron is shown below. Simply add this line to the crontab file for the root user:

```
0 5 * * * cd /usr/openv/netbackup/vault/production; ./bpvault.all dup_param
```

The example above would run a bpvault session every day at 5:00am, using the vault named "V1", the parameter file named "dup_param", and generate the standard bpvault reports.

Maintenance

Cleanup of Bpvault Related Files

Bpvault does not automatically remove old or unused files. It is the decision of the system administrator as to when the files in the bpvault directories should be removed. Bpvault does not reference the files in the session working directories after the session is complete, unless an administrator wishes to reprint any reports or re-send any eject commands to a robot. Many sites use cron to remove files older than a given number of days. It is recommended that the system administrator use bpvault for a few days and determine the disk usage associated with each bpvault session. Then, determine how much historic information would be suitable for the system configuration. Many sites are removing bpvault files if they have not been modified in thirty (30) days. An example of how this could be accomplished with cron is listed below:

```
0 6 * * * find /usr/openv/netbackup/vault/V1 –mtime +30 –exec rm –f {} \;
```

This would remove all data files in the vault working directories if their file modification time is older than thirty (30) days ago; cron would run this command every day at 6:00am.

General Maintenance

For the most part, bpvault is a batch processing program that maintains itself. Most maintenance that a system administrator will do involves cleaning up files that are no longer useful (see "Cleanup of Bpvault Related Files"), making sure there is enough disk space available in the partition on which the directory /usr/openv/netbackup/vault resides, and occasionally modifying the parameter files.

Most parameter file modifications involve changing the class names; changing the "drive_pairs", "server" and "dstunit" entries as more resources become available to the NetBackup configuration; or modifying the "duplicate_days" and "days_startfrom" parameters in case of a duplication failure on a given day. Most of the other parameters should not change after the initial installation.

Installation Summary

To sum up, the following procedures are required to install bpvault on the NetBackup master server:

1. Obtain the bpvault package from the VERITAS Consulting Services Group (this package will most likely reside on an ftp server).
2. Unpackage the bpvault distribution file using uncompress, tar and crypt into the /usr/openv/netbackup/vault directory.
3. Create the symbolic link to your appropriate hardware platform for the bpvault binary.
4. Setup any necessary volume pools, classes, schedules, volume groups, etc. for bpvault.

5. Setup appropriate parameter files for your site configuration. This includes all parameter files for all vaults (e.g. multiple TLD robots). Modify each parameter file according to your offsite requirements (classes to duplicate, days to search in the NetBackup database, volume pool to use for duplication, robot configuration, destination storage unit configuration, NetBackup database backup configuration, report configuration, debugging configuration, etc.).
6. Modify "/usr/openv/netbackup/vault/production/bpvault.env" for your specific site environment (printer names, email addresses, etc.)
7. Modify "/usr/openv/netbackup/vault/production/bpvault.reports" if you do not wish to generate the default set of reports.
8. Modify "/usr/openv/netbackup/vault/bin/eject_tapes" (for ACSLS robots) and/or "/usr/openv/netbackup/vault/bin/robot_inventory" (for all robots) for your specific robotic configuration. This includes modifying hostnames, robot numbers, etc.
9. Modify "/usr/openv/netbackup/vault/production/bpinject" (for TLD robots) to be sure that OS specific environment variables (e.g. RSH, GREP) are configured properly.
10. Setup any remote host configuration. For an ACS environment, this includes setting up a .rhosts file for the user "acsss" on the ACSLS server to allow the root user on the NetBackup master server to login without a password. This access is required for automatic tape ejection and robotic inventory. For a TLD environment, this includes setting up a .rhosts file for the root user on each NetBackup server that controls a TLD robot to allow the root user on the NetBackup master server to login without a password. This access is required for automatic tape injection (see the discussion of the bpinject script in the section entitled "Other Configuration Topics").
11. Setup your NetBackup master server to automatically execute bpvault at specific time periods. This can be done using the system scheduler, cron, or a third-party scheduling utility (e.g. Control-M, AutoSys, etc.). For an example of how to use bpvault with cron, see the discussion in the section entitled "Execution and Scheduling". If you decide not to setup bpvault to run automatically, an administrator must login to the NetBackup master server and execute bpvault manually as described above.
12. Execute the main bpvault program (through bpvault.all) with some test parameter files to get the feel for how bpvault works. Examine the output files (e.g. bpvault.all.output, log.file, etc.) and verify the operation by looking at the reports that are generated.

Procedures – Vaulting

This section outlines the steps to run the bpvault system for duplication of images and vaulting of original images. Normally, these commands are executed in the order that they are documented below. For further details of the order of execution, please see the script in "/usr/openv/netbackup/vault/production/bpvault.all".

This section should be used as a reference guide. It describes each step of the process that is used within bpvault.all.

New Session

bpvault -function id -new_dup_id

This command is used by "bpvault.all" to create a new session number and working directory for the subsequent commands and log files. Even if you are using bpvault for original media ejection, a new session ID needs to be created to store all of the appropriate working files.

Preview Mode

bpvault -dup_id $DUPID -function preview

The primary method bpvault uses to inventory images is to first run bpduplicate as a "preview" mode. For each class configured in the parameter file, bpduplicate looks up the images in the NetBackup image catalog and determines whether or not the image has already been vaulted. It uses the "duplicate_days" parameter to determine how far back to look in the catalog. The use of "duplicate_days" allows us to miss a production day and still find images that need to be vaulted. Seven (7) days is a typical window that ensures retries for duplicating images missed or which may have had media missing or in use errors. There is also a "days_startfrom" parameter that can be used to narrow the search. The "duplicate_days" parameter defines the furthest number of days to search in the past FROM TODAY. The "days_startfrom" parameter defines from which day to start the search.

For example, if today was Tuesday and you wished to preview all images that were written over the weekend (from Friday to Sunday), you would need to setup the parameters as follows:

```
duplicate_days 4
days_startfrom 2
```

These parameters state that we should look back as far as 4 days ago (Friday), but start searching back from 2 days ago (Sunday). Therefore, we are only looking at images whose dates are in the range of Friday to Sunday.

The preview mode places all the images found into a file, "dup.preview.out", which is stored in the working directory for this duplicate session. The status of running the preview mode is found in the "bpvault.all.output" log file.

It is recommended to run a preview mode every time you modify the parameter file to verify the correctness of your changes. This can be done by using the "bpvault.menu" script to execute the preview mode WITHOUT executing the duplication, tape ejection, etc. Once you execute the preview mode, you can examine the file "dup.preview.out" to see what images would be selected if you were to use this parameter file.

Filter Mode

bpvault -dup_id $DUPID -function filter

Bpvault can also be used to offsite the original media on which the backups were written. To do this, it is required to use the appropriate parameters (see the Appendix below describing "vault_type") in the parameter file. Bpvault will require that a preview mode be executed prior to the filter, since the filter will generate a subset of the preview mode images. It is also required that some image and media inventories be executed prior to using this option. Please see the bpvault.all script for an example.

The filter mode will check the inventory of the Media Manager database and determine if any tapes specified in the image list (generated by the preview mode) have already been moved offsite. If so, bpvault will filter out these images and only eject the tapes that are currently in the volume group specified by the parameter file (see parameter "robot_group"). This mode will also filter out any images that have been written to any media which have a last written time younger than days_startfrom. This will guarantee that newer media will not be vaulted accidentally.

NOTE: If you notice that the filter mode is not producing any images to be processed, it is likely that the media to be ejected are not currently in the "robot_group", or in the volume pool(s) specified in the parameter "pool".

Duplication Mode

bpvault -dup_id $DUPID -function duplicate_bymid

This mode will group the images listed in the preview file ("dup.preview.out") by their media ID and then duplicate the images in a batch. This mode allows for bpduplicate to avoid extra mounts and dismounts of media, which can cause significant delays in duplication time.

The "duplicate_bymid" mode will organize the images by media ID. After sorting by media ID, it will start a process for each drive pair for each server that will begin to pick out each image from the respective "bpvault.dup.N" file, where **N** equals the drive_pair counter for that server. The images are sorted out by their server to ensure that duplicates are not done across the network and then split into the number of drive pairs on that server. The total number of "bpvault.dup.N" files equals the number of drive pairs. The total number of images in these files equals the number of images listed in "dup.preview.out". If the process determines that the next image in the list is located on a different media ID than the current, it will collect all images on the same media ID and schedule them for duplication as a batch using a special feature within bpduplicate. During the batch duplication, both the source and destination media ID will remain mounted (unless tape spanning occurs). After all of the images within a given batch have been duplicated, the process will begin reading the "bpvault.dup.N" file for more images until the file has been exhausted.

The status of all the duplicates is found interleaved together in the "bpvault.all.output" file. During actual processing of a specific image, the bpduplicate log output is also found in "bpvault.error.N" log file. Bpvault automatically monitors the progress of each bpduplicate process and waits until it is completed before starting the next bpduplicate.

This mode also includes an option which sorts by media ID in ascending and descending order prior to duplication to avoid media contention on servers that are attached to more than one drive pair.

Suspend Mode

bpvault -dup_id $DUPID -function suspend_media

NOTE: THIS MODE IS ONLY VALID FOR VAULTING ORIGINAL MEDIA.

This mode will freeze or suspend original media within the NetBackup databases after vaulting has occurred.

To invoke this mode, the following parameter must be defined in the parameter file:

```
suspendmedia 1
```

This will enable the suspend mode for the session when executing bpvault.all. The other parameters that must be defined are:

```
bpmedia_command
bpmedia_option
days_suspend
```

The “days_suspend” parameter should be set to the number of days PRIOR TO today that you wish to freeze/suspend media. The following date range is used when determining which media is frozen/suspended:

days_startfrom - number of days FROM TODAY to set the START date

days_suspend - number of days FROM TODAY to set the END date

(days_suspend should ALWAYS be less than days_startfrom)

So, if today is 1/30/99 and you set:

```
days_suspend 5
days_startfrom 10
```

Then bpvault will suspend media found between 1/20/99 to 1/25/99.

For further information on the other parameters, please see the Appendix.

To determine which media were expired, you can then view the output file (normally, “bpvault.all.output_$VAULT” to see which media were expired, or generate the "Media List" report from within the NetBackup GUI.

Duplication Performance

Power Tip

Use of excessive multiplexing will slow down duplication speed. This is because NetBackup must de-multiplex the backup image before copying it to the duplication tape. Reducing the maximum number of multiplexed clients also reduces the number of tapes any specific image is stored on for normal backup and thus reduces the amount of tape needed to read the entire image.

Procedures – Monitoring

This section outlines procedures to setup appropriate monitoring for bpvault.

Monitoring Batch Output File

The file **bpvault.all.output_$VAULT** shows the status of all the various bpvault commands. It is erased and re-created each time bpvault.all is run. The file is initially located in the "/usr/openv/netbackup/vault/production" subdirectory, then copied to the working directory for the session once it is complete. Since you can run more than one bpvault.all script at the same time, the name of the vault is appended to this output file.

Generally, this file contains bpvault and bpduplicate output. This output is merged for all of the drive pairs during the duplication mode.

Additional logging information can be obtained by turning debugging on in the parameter file (see parameter "debug"). Debugging output shows primarily the internal way bpvault code is running.

See the Appendix for a sample output file.

Monitoring Log File

The file **log.file** shows the status of each image that is duplicated and it is located in /usr/openv/netbackup/vault/$VAULT. It is designed to state whether each duplicated image is "ok" or "error". This file can be easily setup to work with an Event Management product to alert the Storage Administrator that a duplication has failed. The line from bpduplicate where the error occurred is also placed into this file to help determine the cause of the failure.

Monitoring NetBackup Log Files

You can monitor NetBackup using the "/usr/openv/netbackup/logs" log files. These require the creation of a subdirectory for the specific NetBackup software you are interested in monitoring. For example, you can monitor the process "bptm" to see the tape manager output. You can create these files on both master and slave servers.

Power Tip

Bpvault can help compare the NetBackup log file output with the bpduplicate log file output. If you enter "command_log" entries into the parameter file, you can see the NetBackup log file entries that are made at the same time of the bpduplicate log file entries. This can be helpful in debugging problems since there is often a time-element to compare. See the Appendix for an example of using bpvault to monitor the NetBackup "bptm" log file.

Bpvault can normally only monitor NetBackup logs on the master server. However, if you wish to also monitor slave server processes, you can do so by NFS mounting their log subdirectories onto the master server, and then adding appropriate command entries in the bpvault parameter file.

For more information about these log files, please see the NetBackup System Administration Guide or the NetBackup Troubleshooting Guide.

Procedures – Media Management

This section outlines steps to track and manage media used for duplication and return.

Determining Volume Inventory

bpvault -function media -command [volume_inv | volume_inv_full | image_inv | media_inv]

This step sets up several files required for media control:

1. volume.inventory: provides an ASCII representation of the Media Manager database for the volume pool used for bpvault duplication and the volume pool used for offsite NetBackup database backups; for original ejects, it provides the same information for all pools listed in the parameter file.
2. volume.inventory.db: provides an ASCII representation of the Media Manager database for the volume pool used for offsite NetBackup database backups.
3. volume.inventory.dup: provides an ASCII representation of the Media Manager database for the volume pool used for bpvault duplication; for original ejects, it provides the same information for all pools listed in the parameter file.
4. image.inventory: provides an ASCII file with NetBackup image catalog information for each image meant to be duplicated.
5. media.inventory: provides an ASCII representation of the media used for both the original image and the duplicated image.
6. volume_full.inventory: provides an ASCII representation of the entire Media Manager database.

These files are normally created by bpvault.all during a session, and are required for other bpvault commands such as tape ejections and report generation.

Assigning Slot ID

bpvault -function media -command assignslot

This step processes the media used file and compares it with the Media Manager database. For each media used in the pool(s) specified in the parameter file, a slot number is assigned. This slot number is unique and is used by the offsite vendor to physically store the tape. When tapes are returned, their slot numbers are zeroed out and reused (see media function “changegroup_torobot”).

It is important to note that this function must be completed in order for any media to be ejected from the appropriate robotic device. Bpvault will not attempt to eject any tapes if a unique slot number has not been assigned to the media. Even if you choose not to use the vault slot ID, this command should still be executed prior to tape ejection.

Changing Tapes to the Vault Group

bpvault -function media -command changegroup_tovault

Each media to be ejected from the robot must be moved into a non-robotically controlled volume group. This step ensures that Media Manager doesn’t try to add more images onto the same tape

and tells NetBackup that the volume is not available for use. The “vault_group” option found in the parameter file is used by bpvault for all ejected tapes in this way. Bpvault reads the media used inventory file and moves each media in the pool(s) listed in the parameter file into this group.

For Media Manager controlled robots (e.g. TLD, TL8), this command is usually not required since the tape eject process will use Media Manager to eject the tapes and change the volume group to a non-robotic group (specified by “vault_group” in the parameter file). If this command is executed prior to tape ejection, Media Manager will move the desired tapes to a non-robotic volume group and assume the tapes have been ejected. If you then attempt to use the Media Manager GUI to eject the tapes automatically, you will be told that the tape does not belong to a robot. Therefore, the desired tapes should be ejected using the media command “eject”.

Ejecting Tapes (using ACSLS)

bpvault -function media -command eject

This step reads the media used inventory file and ejects each tape that is (1) in the pool(s) listed in the parameter file (see the parameter “pool”), (2) in the robotic volume group (see the parameter “robot_group”), and (3) assigned a unique offsite slot number (see the media command “assignslot”). The actual eject is done by an external shell script specified in the parameter file (see the parameter “eject_command”). Bpvault sends several eject commands to the shell script which then forwards them to the appropriate robotic command. A log file “bpvault.eject.log” is created for the session.

The commands generated with this option are specific to ACSLS (e.g. eject 0,0,0 followed by volume serial numbers) and are written to the file specified by the eject_command parameter. The ACSLS commands that are sent to the ACSLS server will be kept in a file “bpvault.eject” in the session working directory (e.g. /usr/openv/netbackup/vault/V1/DUP26). You may re-send these commands to the ACSLS command processor (cmd_proc) if the eject fails for any reason. This option is available in the operations menu (bpvault.opsmenu; see the Operations Guide for more information).

Ejecting Tapes (using TLD, TL8, TLH)

bpvault -function media -command eject

This option is similar to the ACSLS “eject” media command except that it generates CLI calls which will automatically eject tapes from the robot. It is important to note that the tapes to be ejected must belong to the appropriate robotic volume group and duplication pool which are defined in the parameter file (see robot_group and pool below), and must be assigned a unique offsite slot number (media command “assignslot”).

The CLI calls are generated and placed in the file /usr/openv/netbackup/vault/$VAULT/DUP$DUPID/eject_tapes. It should be noted that bpvault does not use the eject_command parameter for the eject command filename. The eject script is placed into the working directory for the session because it could then be re-executed in case of a robotic failure during the initial tape ejection. The script that is generated will also pause after all export slots have been filled and await input from the keyboard. The parameter file should specify how many export slots should be used and whether or not the pause feature should be disabled (i.e. to allow for automatic tape ejection from a scheduler such as cron) by using the parameters “tl*_count” and “robot_eject”, respectively.

Receiving Tapes

bpvault -function media -command robot_inventory

bpvault -function media -command changegroup_torobot

These two commands are used by bpvault.all to find media returned to the silo. The process requires the “Picking List from Vault” report, which is sent automatically to the vault vendor for tapes that have fully expired. Once the tape is expired, the report is generated and a “date requested” field is placed in the description field. (When a tape goes offsite, the “date requested” that is placed into the description field is noted as 00000000). When the tape is entered into the robot, bpvault will automatically recognize it from the robot inventory. It then changes the volume group to the “robot_group” (specified in the parameter file) to allow the tape to be reused. This step also zeroes out the slot number and zeroes out the date requested.

For non-ACSLS controlled robots, the second command (changegroup_torobot) should be executed, but may generate some “error” information in the log files. This is because changegroup_torobot will check the volume group of each media and only change the volume group if the tape is currently in the “vault_group” and is destined for the “robot_group”. However, the important aspect of changegroup_torobot that will be used is that of zeroing out the slot number and date requested fields. See the bpvault.tl*.eject scripts for the proper placement of these commands.

NOTE: the media command “robot_inventory” will execute the script “/usr/openv/netbackup/vault/bin/robot_inventory”; this script should have been customized for your site configuration during the installation process.

Ejecting Original Media

bpvault -function media -command eject_orig

Ejecting original media is different from vaulting original media (see “Other Configuration Topics”) because ejecting original media does not assign an offsite slot number to the media, nor does it enable automatic retrieval of media from an offsite location. This process is normally used to easily move media from the silo(s) to an onsite storage location. One reason for ejecting original media from the silo(s) would be to free up slots within the silo(s) for new tapes.

To eject media for a given duplication session, the bpvault.menu script should be used, which executes this media command. The option “Inventory, Eject Originals” should be selected and will determine which original media was used during the session. Bpvault will then generate a series of eject commands for this media depending on the type of robotic control defined in the parameter file (see parameter “robot_type”).

It is also recommended to generate one of (or both of) the Original Media Eject List reports, which will document which media has been ejected. See the “Procedures – Printing Reports” section for more information about these reports.

Checking Robot/Media Status

For most robots, Media Manager can be used for most operational duties, such as requesting an inventory of the robot. It is always recommended that a storage administrator use the option “Use Inventory to Update Volume Configuration” (under the “Robots” menu option) in the Media Manager GUI to periodically verify that the Media Manager database is synchronized with the robotic controllers.

Procedures – Printing Reports

This section describes the different reports that are available within bpvault and how to access them from the command line.

There are four categories of reports generated by bpvault

- media that are going offsite to the vault
- media that are coming onsite from the vault
- detailed information about the media
- special reports for ejected original media

Each of the reports are listed below in terms of which category they belong to.

NOTE: Report commands with a suffix of "_ff" are a flat file version of the particular report. The flat file versions of these reports contain no header information and display information with fixed field lengths. If your site is currently using a different database system to track tapes, these reports can be imported using the fixed field lengths.

NOTE: The tapes used for NetBackup database backups will not show an "ASSIGNED" date, but rather display NBDBTAPE. The "EXPIRATION" date for these tapes will be that which is calculated as a return date during the assignment (see media command "assigndb_dup"). It also should be noted that the "#IMAGES" and "KBYTES" fields will display zero (0) because NetBackup is not aware of how much data was backed up during the NetBackup database backup. For more information, see the section entitled "Procedures – Vaulting NetBackup Databases".

Media Destined for Vault

bpvault -function report -command [picking_library | picking_library_ff]

bpvault -function report -command [dist_vault | dist_vault_ff]

These reports show the tapes that have been ejected from the silo and should be transported offsite. The Picking List from Library report (picking_library) is sorted by media ID (also known as VOLSER) and should be used by the operations staff as a checklist for media that has been ejected from the silo(s); the report can be saved for tracking purposes. The Distribution List for Vault report (dist_vault) is sorted by offsite slot number and should accompany the media that is destined for the offsite vault. It should be used by the vault vendor to verify that all the tapes reported were actually received.

Media Returning from Vault

bpvault -function report -command [dist_library | dist_library_ff]

bpvault -function report -command [picking_vault | picking_vault_ff]

These reports show the tapes that are being requested back from the vault. Tapes are listed on this report because bpvault determined that they are (1) in the appropriate offsite volume group ("vault_group" in the parameter file) and (2) expired from NetBackup (i.e. no longer containing valid NetBackup images). Once bpvault finds these tapes, it changes the "date requested" field within the Media Manager description field for the media. It then prints out this media ID

(VOLSER) on this report along with the date requested. Both reports are identical except for the report header. One copy of these reports (Picking List for Vault) should be sent to the vault vendor, while one copy should remain onsite to be used as a checklist for when the media return from offsite.

Detailed Media Reports

bpvault -function report -command eject_detail

This report is similar to the Distribution List to Vault/Picking List from Library report (dist_vault/picking_library), except that detailed information is listed for each media. This detail includes the client machines that have backups on this media, when the client was backed up, the NetBackup backup identifier for the backup job and the number of kilobytes stored in this NetBackup fragment. Keep in mind that backup jobs could span multiple tapes, so it is possible to see duplicate detailed listings of a given backup job on more than one tape. Also, true image recovery (TIR) information will be listed on this report and will be noted by a “TIR” adjacent to the number of kilobytes. This report is very useful at a disaster recovery site and it is recommended to send this report offsite.

bpvault -function report -command eject_detail_sum

This report is similar to the Detailed Distribution List report (eject_detail), except that each media will list only a unique client, class, schedule and date. That is, if multiple backup jobs for a given client, class and schedule (usually seen with RDBMS backups or SAP backups) are written to the same tape on the same date, only one line of information will be printed out on this report. The Detailed Distribution List would show each of these backup jobs as a separate entry, which may generate a very long report. This report summarizes the information and presents it in a more compact form. This report is also very useful for disaster recovery situations and it is recommended to send this report offsite.

bpvault -function report -command [vault | vault_ff]

The Vault Inventory report shows all media that are currently offsite at the vault vendor AFTER the current batch of tapes have been processed and sent offsite. This list of tapes is generated by checking the description field for the media, the volume pool (specified as “pool” in the parameter file) and the volume group (specified as “vault_group”). If any tapes have been requested to be returned from the vault vendor, they will not be displayed on this report (see vault_full report below). This report is a “sanity check” for the vault vendor to verify that bpvault and the vault vendor agree on the tapes currently offsite. It is recommended to send this report to your vault vendor to perform this verification.

bpvault -function report -command [vault_full | vault_full_ff]

This report is similar to the Vault Inventory report (vault) except that it includes any tapes that have been requested back from the offsite vault vendor. Normally, this report is not generated on a daily basis since the Vault Inventory report is usually sent to the vault vendor to perform verification.

bpvault -function report -command inventory

This report shows all tapes within the vault pool (specified by “pool” in the parameter file) volume pool. If the tapes are currently at the vault, the code “V” is used under the “Location” designation while the code “R” is used for tapes that are currently within the robot. It is imperative that tapes within the vault pool belong to either the “vault_group” or “robot_group” volume group, since this report relies on those volume groups.

bpvault -function report -command recovery

This report shows all classes defined on the NetBackup master server and all media that is required for restores between a given set of dates. The header of the report denotes which date ranges are covered by the report. These dates are specified in the parameter file by the parameters “recoverto_days” and “recoverlength_days”. The default values for these parameters is 0, so it is recommended to define them appropriately to ensure an accurate report. The first parameter, “recoverto_days”, defines the number of days before today which is used as the end date for the report. The second parameter, “recoverlength_days”, defines the number of days BEFORE THE END DATE which is used as the start date for the report. For example, if the following was defined in the parameter file:

```
recoverto_days 1
recoverlength_days 14
```

And, today is January 30, 1998, then the report would encompass an end date of January 29, 1998 (1 day before today) and a start date of January 15, 1998 (14 days before January 29, 1998). Most users will define “recoverto_days” as 0 (which means the report would have an end date of today) so that they can determine which tapes will be needed to recover to today. This report is very useful at a disaster recovery site if only a handful of tapes are necessary for recall.

This report also includes the NetBackup database tapes that are currently offsite in the respective vault. For the NetBackup database media to be listed in this section, their volume group must match the volume group specified in the “vault_group” parameter. Also, only NetBackup database media that are assigned (i.e. valid) will show up on this report.

Special Reports

bpvault -function report -command eject_orig

This report should be run only after ejecting original media. This is not the same as vaulting original media (using “vault_type” parameter), which assigns offsite slot numbers to tapes and tracks the tapes for returning from the vault. See the “eject_orig” media command, and the “Eject Originals” option in bpvault.menu. This report shows the media ID (VOLSER), assigned date and expiration date of all original media used during a given duplication session after it has been ejected.

bpvault -function report -command eject_orig_detail

This report is similar to the Original Media Eject List report (eject_orig) and includes a detailed inventory of each media, including clients, backup times, backup ids and size of the backup. The detailed information in this report is similar to the Detailed Distribution List report (eject_detail). This report should only be generated after original media for a given session have been ejected (see the “eject_orig” media command, and the “Eject Originals” option in bpvault.menu).

Procedures – Vaulting NetBackup Databases

Vaulting of NetBackup Database Media

Bpvault is capable of backing up the NetBackup and Media Manager databases to pre-assigned tapes and include these tapes in the standard reports. These database tapes will also be called back from the vault after a specified period of time.

To enable the NetBackup database backups, the following parameters must be defined in the parameter file:

```
num_dbdups 1
days_holddbtape 7
dbpool NBDB_Duplicate
```

The parameter “num_dbdups” defines how many copies of the NetBackup database should be made. If this parameter is zero (0) then no tapes will be created with NetBackup database information on them, and the NetBackup database backup procedure (function “duplicate_db”) will be skipped.

The second parameter, “days_holddbtape”, defines how many days (starting from the day that the command “assigndb_dup” was issued) in the future we should hold the database tapes. This date is calculated and placed in the “date requested” field of the media description field.

The third parameter, “dbpool”, defines the name of the pool used for NetBackup database backup duplicates. This pool MUST BE DIFFERENT from any other pool(s) listed in the “pool” parameter(s).

Optionally, the administrator may specify path names to be backed up during the NetBackup database backup. If no paths are defined in the parameter file, then the paths specified in the “Change NetBackup DB Backup Attributes” (within the NetBackup GUI) are used. To specify paths within the parameter file that are DIFFERENT from those in the NetBackup interface, use the following parameters:

```
dbbackuphost server-name
dbpath master:/usr/openv/netbackup/db
dbpath master:/usr/openv/volmgr/database
dbpath slave:/usr/openv/netbackup/db/media
dbpath slave:/usr/openv/volmgr/database
```

The above example tells bpvault to use “server-name” to backup the NetBackup databases (in case the master server does not have a locally attached tape drive), and to backup four distinct directories residing on the servers “master” and “slave”. The path name syntax is identical to that specified in the NetBackup System Administration Guide (see “NetBackup Database Backups”). The path names specified with the parameters “dbpath” will override those specified in the NetBackup GUI.

NOTE: If the NetBackup master server does NOT have any locally attached tape devices, then the above parameters “dbbackuphost” AND “dbpath” MUST be defined.

The commands required to vault the NetBackup database media are used within the standard bpvault.all script. To understand what they do, they are listed below:

bpvault -function media -command assigndb_dup

This media command will assign a specified number of tapes (see “num_dbdups” parameter) from the dbpool volume pool (see “pool” parameter) that are currently unassigned. If the number of tapes requested is larger than the number of tapes available in the volume pool, only the number of available tapes in the pool will be assigned. Also, the tapes to be assigned are required to belong to the bpvault volume group (see “robot_group” parameter). At the time of assignment, bpvault will place an "expiration date" of the database tapes (see “days_holddbtape” parameter) in

the description field, which is normally used for the “date requested” field for regular bpvault tapes. When the Picking List for Vault/Distribution List for Library report is generated, bpvault will determine if any NetBackup database tapes should be recalled from the vault based on this date. After executing this media command the Media Manager database should be re-inventoried (see media command “volumeinv”) in order to obtain any changes made. This media command should be executed prior to the media commands “image_inv”, “media_inv” and “assignslot”. See the script “bpvault.all” for the proper order of commands.

bpvault -function duplicate_db

This bpvault function will initiate a backup of the NetBackup and Media Manager databases using the command line interface bpbackupdb. It is important to note that this function will use the path names defined with the “Change NetBackup DB Backup Attributes” option from the NetBackup GUI, unless you specify the appropriate paths within the parameter file (see “dbbackuphost” and “dbpath” in the parameter file). Bpvault will mount the tapes assigned with the “assigbdb_dup” media command, backup the database paths specified by the administrator and unmount the tapes. If multiple copies are specified (see “num_dbdups” parameter), the database backups will occur serially (one by one).

Finally, it should be noted that if a NetBackup database backup is specified using the above parameters and there are enough tapes to allow assignment, then these tapes will appear on the Picking List for Library, Distribution List for Vault, Detailed Distribution List, Summary Distribution List and Vault Inventory reports. The tapes used for NetBackup database backups will not show an "ASSIGNED" date, but rather display NBDBTAPE. The "EXPIRATION" date for these tapes will be that which is calculated as a return date during the assignment (see media command “assigndb_dup”). It also should be noted that the “#IMAGES” and “KBYTES” fields will display zero (0) because NetBackup is not aware of how much data was backed up during the NetBackup database backup. To verify that a NetBackup database backup is valid, use the command “/usr/openv/netbackup/bin/admincmd/bprecover” which is documented in the NetBackup Troubleshooting Guide, Chapter 5.

Manual Deassigning of Vaulted NetBackup Database Tapes

To be able to remove the NetBackup database tapes from bpvault control, it is necessary to deassign each tape. The following command should be used with caution:

```
/usr/openv/volmgr/bin/vmquery -deassignbyid <media-id> <pool-num> 1
```

In the above command, <media-id> is the media ID that is to be deassigned, and <pool-num> is the pool number that <media-id> belongs to. To obtain the pool number, use the “vmquery” command as in the example below:

```
# /usr/openv/volmgr/bin/vmquery -m S04440
================================================================================
media ID:              S04440
media type:            8MM cartridge tape (4)
barcode:               -------------
description:           CH_V1|101|S278|00000000
volume pool:           Duplicates (3)
robot type:            NONE - Not Robotic (0)
volume group:          vault_grp
created:               Tue Sep 3 10:08:32 1996
assigned:              Tue May 6 00:11:45 1997
last mounted:          Tue May 6 11:34:25 1997
first mount:           Tue Sep 3 18:20:48 1996
expiration date:       ---
number of mounts:      21
max mounts allowed:    ---
status:                0x0
================================================================================
```

The pool number is listed on the line “volume pool”, and is the number in between the parentheses. In this case, the media ID would be S04440 and the pool number would be 3.

Procedures – bpvault.menu

This section documents how and when to run commands from the bpvault.menu

Bpvault.menu Overview

The bpvault.menu is designed for use by the Storage Administrator to manually run specific commands in case the bpvault.all script is interrupted (e.g. by system crash or halt).

The command line usage for the bpvault.menu script is:

```
cd /usr/openv/netbackup/vault/production
./bpvault.menu <parm file>
```

where:

```
<parm file> is the parameter file
```

The menu below would be seen by typing: ./bpvault.menu dup_param.

If you are vaulting duplicated images (“vault_type” is defined as “normal” in the parameter file), then the menus will appear as the following:

If using a robot that is not controlled by Media Manager (e.g. ACS, TLH), the menu looks like:

```
bpvault menu

Current Vault:  V1
Current Session: DUP 86
Parameter:  dup_param
Output:  bpvault.menu.output

1)  Change Vault
2)  Change Session Number
3)  Change Parameter File
4)  Change Output File
5)  Create New Dup Session
6)  Inventory Media and Detect Returned Media
7)  Run Preview Mode
8)  Run Duplicates
9)  Inventory Volumes, Images and Media
10)  Assign Slot IDs to Offsite Media, Backup NBU DB
11)  Inventory, Change Offsite Media to Vault Group
12)  Inventory, Eject Offsite Media (Originals or Duplicates)
13)  Inventory, Eject Onsite Media (Originals)
14)  Change Expiration Dates of Duplicates
x) Exit
```

If using a Media Manager controlled robot (e.g. TLD, TL8), the last few options would look like:

```
10)  Assign Slot IDs to Offsite Media, Backup NBU DB
11)  Inventory, Eject Offsite Media (Originals or Duplicates)
12)  Inventory, Eject Onsite Media (Originals)
13)  Change Expiration Dates of Duplicates
x) Exit
```

If you are vaulting original images (“vault_type” is defined as “original” in the parameter file), then the menus will appear as the following:

If using a robot that is not controlled by Media Manager (e.g. ACS, TLH), the menu looks like:

```
bpvault menu

Current Vault:  V1
Current Session: DUP 86
Parameter:  dup_param
Output:  bpvault.menu.output
```

```
1)  Change Vault
2)  Change Session Number
3)  Change Parameter File
4)  Change Output File
5)  Create New Dup Session
6)  Inventory Media and Detect Returned Media
7)  Run Preview Mode
8)  Inventory, Filter for Originals
9)  Inventory Volumes, Images and Media
10)  Assign Slot IDs to Offsite Media, Backup NBU DB
11)  Inventory, Change Offsite Media to Vault Group
12)  Inventory, Eject Offsite Media (Originals or Duplicates)
13)  Suspend Original Media Based on Search Criteria
x) Exit
```

If using a Media Manager controlled robot (e.g. TLD, TL8), the last few options would look like:

```
10)  Assign Slot IDs to Offsite Media, Backup NBU DB
11)  Inventory, Eject Offsite Media (Originals or Duplicates)
12)  Suspend Original Media Based on Search Criteria
x) Exit
```

Change Vault, Session, Parameter Files

Normally, the menu will show the current vault, session and parameter file in use at the site. If you use this default setup, you will be running these steps for the most recent session. In the case of a system reboot or other failure of bpvault.all, this is the correct setup. There normally shouldn't be a reason to re-run steps for other sessions. Also, you shouldn't have bpvault.all running for this vault while using this menu. The issue is that the Media Manager database changes after each duplication session and it is possible to create errors in the system. These errors could occur, for example, if slot numbers were assigned simultaneously in the same vault by two different bpvault scripts. So, the primary purpose of bpvault.menu is to complete a previous session prior to starting another session for the same vault.

Change Output File

Output from the bpvault.menu is a different file than the normal bpvault.all.output file. This will allow you to confirm the processing of the menu jobs. Note that the output file is not erased - you may wish to do this manually before running any job here.

Create New Dup Session

This is the first step run by bpvault.all. Normally, you will not use this step unless you are trying to test the bpvault program or manually running a full duplication session.

Inventory Media and Detect Returned Media

This step inventories the robot and updates the Media Manager database for any media that has returned from the vault.

Run Preview Mode

This step searches for classes and images to vault. Normally, you will not use this step unless you are manually running a full session, or you are testing a new parameter file. You may wish to review the "dup.preview.out" file (in /usr/openv/netbackup/vault/$VAULT/DUP$DUPID) to determine which images would be duplicated.

Run Duplicates

This is the step that creates the duplicates. You will use this step for manual duplication only. While this step is running, you may wish to monitor the output file for progress.

Inventory, Filter for Originals

This step is performed only for the vaulting of originals. This process will determine which media has already been vaulted and filter out any images which have been written to that media. It will also filter out any media which does not satisfy the date range requirement (see “Procedures – Vaulting, Filter Mode”).

Inventory Volumes, Images and Media

This step inventories all the images that were processed during a session. If the duplication process was aborted for some reason (system crash, etc.), it is likely only a partial set of images were duplicated. This step will build data files which are used by other bpvault commands, such as tape ejection.

Assign Slot IDs to Offsite Media, Backup NBU DB

This step assigns the used media specific slot numbers in the vault. This process must be completed before any tapes can be ejected from robotic libraries.

If the parameter file specifies that a backup of the NetBackup databases should occur (“num_dbdups” defined as greater than 0), then this will occur after assigning the offsite slot numbers.

For robots not controlled by Media Manager (ACS, TLH):

Inventory, Change Offsite Media to Vault Group

For robots not controlled by Media Manager, the Media Manager volume group must be altered separately from the physical eject process. This is because Media Manager does not support ejection of media from certain robots. Once this step is complete, Media Manager will think that these tapes are no longer robotically controlled. The volume group used here is specified in the parameter file (parameter “vault_group”). Bpvault will determine which tapes qualify for vaulting and change their volume group to the non-robotic volume (vault) group.

Inventory, Eject Offsite Media (Originals or Duplicates)

This step re-inventories the media used and ejects them from the robot. This is accomplished by generating an eject command for each tape, outputting the commands to a shell script and executing the script. If only some tapes were ejected during a normal run, you may also use this step to re-send the same eject commands to the robot. For example, if the silo doors were open during the eject process, the command may not have completed successfully so you may need to re-send the eject commands.

For Media Manager controlled robots:

Inventory, Eject Offsite Media (Originals or Duplicates)

This step is similar to the eject option for robots that are not controlled by Media Manager, except there is no need to manually move the media to the vault group. The reason for eliminating this step is because Media Manager can automatically eject tapes from the robot and update their volume group in one step. Therefore, the bpvault command “changegroup_tovault” is not used for Media Manager controlled robots.

This step will inventory the Media Manager database and eject tapes from the robot. A shell script with vmchange commands will be generated and written to

/usr/openv/netbackup/vault/$VAULT/DUP$DUPID/eject_tapes. This shell script will then be executed and should automatically place the appropriate tapes into the robot's export slots.

For all robots:

Inventory, Eject Onsite Media (Originals)

If you are vaulting duplicates ("vault_type" defined as "normal" in the parameter file), this step will enable you to eject the original media used during this duplication session. For example, you may wish to use this option to transfer original media from the robotic silo to an onsite safe at given intervals. Bpvault will read the media used file and the full volume inventory, and determine which media contains the original backup images. Then, depending on the type of robotic control used, bpvault will eject the appropriate media from the silo.

Change Expiration Dates of Duplicates

This step will update the expiration dates of the duplicate media based on the "retention_length" values in the parameter file. This will allow you to have different expiration dates for your original and duplicate media. This option is only valid for duplication sessions (e.g. "vault_type" set to "normal"). For more information on this option, please see the "New Features" section.

Suspend Original Media Based on Search Criteria

This step will suspend original media based on the "days_suspend" parameter. This option is only valid for original media sessions (e.g. "vault_type" set to "original"). For more information on this option, please see the "Procedures – Vaulting" section, subheading "Suspend Mode".

Reports

The script "bpvault.opsmenu" script can now be used to print out the vaulting reports. You may need to send the reports by email as well. See the Operations Guide on how to use this menu.

Procedures – Recovery

This section provides information on how to use the duplicated images for restoring. This is currently a manual process for bpvault. An example shell script is listed at the end of this section, which can be used to automate part of this process for a disaster recovery.

Tape Recovery vs. Disaster Recovery

There are several possible reasons for using the duplicated images for recovery. If there is a large failure, e.g. loss of the tape silo, then one needs to follow a disaster recovery plan. That could include rebuilding the NetBackup software and NetBackup image catalog, determining the state of the catalog vs. the existing backups (e.g. loss of catalog information since last backup), rebuilding the catalog by reading images on tape, using duplicated tapes and finally restoring actual images. That level of problem is beyond the scope of this document as it involves other complicated steps and procedures of the basic NetBackup software. Rather, this section discusses the simple step of how to use duplicated images for restores and assumes that the NetBackup system and image catalog are current and up to date.

Determining Tape or Tapes to Use for Recovery

The basic reason to use a duplicated tape is that the original backup tapes are lost or damaged for a specific image or set of images. That is, there was an attempt to restore from a tape and an error occurred, either a read error from the tape or an inability to find the tape. The steps to follow then are:

1. Identifying the damaged tape.
2. Determining which images were originally on the damaged tape.
3. Determining which duplicate tapes were made for these images, if any.
4. Telling NetBackup to use the duplicate for restore of these images.
5. Freezing the tapes containing the duplicated versions so they won't expire.
6. Requesting the tapes to be returned from the offsite vendor.
7. Placing the tapes back into the silo and notifying Media Manager of new location.
8. Performing normal restore, which will automatically request the mount of the duplicated versions.
9. Unfreezing the duplicated tapes to allow the image/media to expire normally.
10. Optionally, making new duplicates to replace damaged ones.

Most commands listed below can be found in the directory /usr/openv/netbackup/bin/admincmd. We recommend that you place this directory in your search path when doing this type of work.

Identifying the damaged tape

This step is normally done by the user requesting a restore and receiving some sort of error message during the restore. Errors are logged in the restore log and also show up on the job monitor as the restore fails (restore monitoring in Job Monitor is seen in NetBackup 3.1 and above). Procedures can be setup using normal NetBackup scripts to send errors to an Event Management console to notify the Storage Administrator immediately of this type of media error. For our purposes, we can assume that this step is done manually by a user or administrator doing regular work.

Determining which backup images were originally on the damaged tape

All images on a specific tape can be identified by running the "bpimmedia" command. This command will scan the NetBackup image catalog so it may take a few minutes depending on the size of that catalog:

```
bpimmedia -mediaid <TAPEID>
```

Here's an example output. It shows that there is one image on the tape "S05423" for the client "fgolddust". It also shows that this image has been duplicated since it has (FRAG 2) entries. The full image name is "fgolddust_0862806643":

```
nirvana# bpimmedia -mediaid S05423
IMAGE fgolddust 2 fgolddust_0862806643 golddust_BR1 0 Full_Weekly 0 3 19360 8654 85043 0 0
FRAG 1 -1 2293 0 2 6 2 S05423 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 1 232848 0 2 6 1 S02643 nirvana 64512 2 862804830 3 0 *NULL*
FRAG 1 2 1225539 0 2 6 2 S02643 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 3 70182 0 2 6 3 S02643 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 4 825700 0 2 6 1 S05423 nirvana 64512 2 862808446 3 0 *NULL*
FRAG 2 -1 2293 0 2 6 2 S04440 nirvana 32768 0 862927577 2 0 *NULL*
FRAG 2 1 2335584 0 2 6 1 S04440 nirvana 32768 2 862927577 2 0 *NULL*
```

Determining which duplicate tapes were used, and their host

In step (2) above, the (FRAG 2) entries show that an image has been duplicated. The (FRAG 2 1) entry is the duplicate copy, whereas on copy 1, there were 4 fragments (usually due to multiplexing). The (FRAG 2 -1) entry is the TIR duplicate. In this case, the image fgolddust_0862806643 is using media "S04440" for duplicating all of the original fragments. This is normal because the original image was multiplexed onto 4 tapes, while the duplicate was de-multiplexed during image duplicating, and could fit on one tape.

Also note that the host for the media is printed for each fragment, in this case "nirvana". With slave servers, the host could be different than the master. Under bpvault, the duplication should normally occur on the same server that made the original backup, so the host server names should be the same for both copies of the image.

You can confirm this information by using "bpimagelist':

```
nirvana# bpimagelist -backupid fgolddust_0862806643
IMAGE fgolddust 0 0 2 fgolddust_0862806643 golddust_BR1 0 *NULL* root Full_Weekl
y 0 3 862806643 4591 865485043 0 0 2356562 19360 2 7 1 golddust_BR1_0862806643_F
ULL.f *NULL* *NULL* 0 1 0 2 865830643 *NULL* 1 0 0 0 0 *NULL*
HISTO -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
FRAG 1 -1 2293 0 2 6 2 S05423 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 1 232848 0 2 6 1 S02643 nirvana 64512 2 862804830 3 0 *NULL*
FRAG 1 2 1225539 0 2 6 2 S02643 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 3 70182 0 2 6 3 S02643 nirvana 64512 0 862804830 3 0 *NULL*
FRAG 1 4 825700 0 2 6 1 S05423 nirvana 64512 2 862808446 3 0 *NULL*
FRAG 2 -1 2293 0 2 6 2 S04440 nirvana 32768 0 862927577 2 0 *NULL*
FRAG 2 1 2335584 0 2 6 1 S04440 nirvana 32768 2 862927577 2 0 *NULL*
```

To confirm which is the Primary Copy (the copy to be used for restores), use the -L option with bpimagelist:

```
nirvana# bpimagelist -L -backupid fgolddust_0862806643 | grep Primary
Primary Copy: 1
```

Telling NetBackup to use the duplicated copy rather than the original

The command "bpimage -npc" is executed to select which image is used for restoring an image:

```
nirvana# bpimage -npc 2 -backupid fgolddust_0862806643 -client fgolddust
```

To confirm the new Primary Copy:

```
nirvana# bpimagelist -L -backupid fgolddust_0862806643 | grep Primary
Primary Copy: 2
```

Freezing the duplicated copy to ensure restore

The command "bpmedia -freeze" is used to keep NetBackup from expiring the images on the media and to keep the media assigned in Media Manager. You should also use the media host for this image, which was printed by bpimage in step (2) above. This is required when the host is different than the machine on which you are running this command.

```
nirvana# bpmedia -freeze -ev S04440 -host nirvana
```

If a restore was initiated at this point, media ID S04440 would be used for the restore.

Requesting the media to be returned from the vault

The process for requesting media from your offsite vendor should be documented and kept with the operations team. In most cases, you will need to fill out a form or contact the vendor by phone to request a particular piece of media to be returned. You will at least need to specify the media ID of the tapes to be used for restores. The offsite vendor may also require an account number, Vault ID and possibly the slot in which the media physically resides. As seen below, you may find most of this information in Media Manager, by using the command vmquery (located in /usr/openv/volmgr/bin).

For example, the slot number can be found by querying the Media Manager and looking at the description field. In this case its S278:

```
vmquery -m S04440
============================================================================
media ID:              S04440
media type:            8MM cartridge tape (4)
barcode:               -------------
description:           CH_V1|101|S278|00000000
volume pool:           Duplicates (3)
robot type:            NONE - Not Robotic (0)
volume group:          vault_grp
created:               Tue Sep 3 10:08:32 1996
assigned:              Tue May 6 00:11:45 1997
last mounted:          Tue May 6 11:34:25 1997
first mount:           Tue Sep 3 18:20:48 1996
expiration date:       ---
number of mounts:      21
max mounts allowed:    ---
status:                0x0
============================================================================
```

Place returned tape(s) back into silo

Once the tape is returned from the offsite vendor, it needs to be placed in the appropriate robotic library. Consult the operations team for details on how to place tapes into the library.

If using a robot not controlled by Media Manager, you will most likely initiate an “enter” command to allow the robot to move the tape from a CAP slot into the library. Once this is complete, you need to inform Media Manager that the tape is no longer non-robotic. You can do this by either selecting the tape in the Media Manager GUI and using the “Move” action (be sure to select the appropriate robot type and select the appropriate robotic volume group), or you can use the “Use Inventory to Update Volume Config” option in the Media Manager GUI.

If using Media Manager controlled robots, you can use either the “Move” command in the Media Manager GUI (discussed above), or use the provided script “bpinject” for TLD robots (located in /usr/openv/netbackup/vault/production). This script scans the export slots of a TLD robot and moves any tapes found into empty library slots. This script must be executed on the robotic control host since the “tldtest” utility is used. The script accepts multiple command line parameters, so be sure to look at the comments at the beginning of the script for proper usage.

NOTE: Some robotic libraries are not able to discern the barcode of a tape in an export slot. Try using the utility “tldtest” from the command line before using bpinject.

Perform normal restore

At this point, a normal restore of the same image should use the duplicate media rather than the original. Monitoring the restore log should show a mount request for the duplicate media. In this example, “S04440” should be used.

Unfreeze media used for duplicates

Once the restore is successful, unfreeze the duplicate media to allow the normal expiration process to be followed. In this case, the media will be re-used by NetBackup for duplication once the backup image has expired.

```
nirvana# bpmedia -unfreeze -ev S04440 -host nirvana
```

Create new duplicate images

See the section “Bad or Missing Duplicate Tape”, under “Procedures – Troubleshooting”.

Modifying the NetBackup Database for a Large Number of Images

It is possible to write a shell script that can automate step (4) above. The following script is an example of how this automation is possible. This type of script could be used in a disaster recovery situation where a large number of images need to have their primary copy modified.

This script is included in “/usr/openv/netbackup/vault/production/changecopy.sh”.

```
#!/bin/sh

#
# changecopy.sh - script that will change the primary copy of a given set
#                 of images to copy 2
#

#
# Where is bpimagelist and bpimage
#
BPIMAGELIST=/usr/openv/netbackup/bin/admincmd/bpimagelist
BPIMAGE=/usr/openv/netbackup/bin/admincmd/bpimage

#
# Select a set of images; depending on which images you wish to modify,
# uncomment only ONE of the commands below and modify the appropriate
# parameters (e.g. classname, schedulename, clientname)
#
IMAGES=`$BPIMAGELIST -d 01/01/70 -idonly | awk '{print $8}'`
#IMAGES=`$BPIMAGELIST -class classname -d 01/01/70 -idonly | awk '{print $8}'`
#IMAGES=`$BPIMAGELIST -class classname -sl schedulename -d 01/01/70 -idonly | awk '{print
$8}'`
#IMAGES=`$BPIMAGELIST -client clientname -d 01/01/70 -idonly | awk '{print $8}'`

#
# For each image change the primary copy
#
for IMAGE in $IMAGES
do
        echo "Changing image $IMAGE primary copy to 2..."
        $BPIMAGE -npc 2 -backupid $IMAGE
done

exit 0
```

Procedures – Troubleshooting

This section discusses potential problems with bpvault and how to work around them. Please be sure to notify VERITAS of these or any other problems to help us improve operations:

Media Missing in Robot

Duplication may fail if a requested duplicate media is not found in the robot. The options “Use Inventory to Update Volume Configuration” and “Robot Inventory” within the Media Manager GUI can be used to compare the tapes actually stored in the robot with the Media Manager database. You must manually fix the problem by determining where the tape is actually stored. Use the Media Manager GUI to move the tape once it has been found. If the tape is in the vault, the volume group should be set to the vault group. If the tape is not found, you should delete it from the NetBackup system. If the tape is missing yet is assigned and has valid duplicate images, you will need to use the command “bpexpdate” (located in /usr/openv/netbackup/bin/admincmd and documented in the NetBackup System Administration Guide) to expire the images before you delete the tape from Media Manager. If using a robot that is not controlled by Media Manager, you may need to use vendor specific command to fix this situation (e.g. the bar code has been removed accidentally from the tape). Refer to robot vendor documentation for more information.

Bad or Missing Duplicate Tape

If a duplicate tape is lost or damaged, you can reduplicate images found on the tape. The steps required are:

1. Determine which images were on the tape. See the Recovery section instructions for this procedure.
2. Expire the lost or damaged duplicate tapes using the command “bpexpdate”.
3. Determine when the images were originally created. Use the command “bpimagelist -U -backupid <backupid>” to see this date.
4. Create a new dup_param file containing **class** entries only for the class names found for these images, and set the **duplicate_days** to be greater than the days shown above. For example, set it to 32 if the original was made one month ago.
5. Use bpvault.menu to manually run necessary duplication steps. Be sure to change the dup_param file name. You’ll need to run “Create Dup Session”, “Run Preview Mode”, “Run Duplicates”, “Inventory Volumes, Media and Images”, “Assign Slot”, “Inventory, Change Duplicates to Vault Group” (for ACS robots only), “Inventory, Eject”. Make sure that the normal bpvault has finished running for the day before running this second instance. Also, you can double check that the lost duplicate images were stored in the “dup.preview.out” file (located in the working directory for this duplication session) before proceeding with “Run Duplicates”.
6. Use the script “bpvault.opsmenu” to print a set of reports for this run. Be sure to provide both sets of reports to the vault vendor.

Need to Stop Bpvault

Power Tip

The only supported way to stop the bpvault duplication process is to use the script “haltdups.now”. This command sends a SIGUSR1 to the main duplication process. This signal is automatically propagated to the other bpvault instances. The current duplication will complete and no more duplicates will run. However, the bpvault.all script will continue to complete processing of the other bpvault commands.

If you wish to “hard kill” bpvault, rather than wait a number of minutes for the current bpduplicate processes to complete, the best method is to first run the “haltdups.now” script, then manually kill the bptm jobs on each server. However, do not kill the bpduplicate processes. The errors received by bpvault will be logged and these images will not be successfully duplicated. The failure of the bptm/bpduplicate will mean these images will be duplicated on the next attempt if the number of duplicate_days is not exceeded.

Permissions Problems

Most of bpvault uses standard NetBackup protection. If using ACS controlled robots, the shell scripts “robot_inventory” and “eject_tapes” will use “rsh” to run. This is a condition based upon current NetBackup support of the StorageTek ACSLS drivers. Be sure the /.rhosts entries contain short and full names for root to perform rsh from the master server to the server running the StorageTek daemons.

Tape Drive or Robot Offline

There may be a problem with ACSLS drives becoming off-line. These will show up on the Media Manager GUI. You should normally try UP’ing the drive and/or Resetting the drive. If drives persistently go offline this may cause the duplication to hang. There may be a problem with keeping track of how many drives are available when duplicates are running.

Also, if the tape drives are listed as “AVR” control in the Device Management GUI, there may be a problem with the robotics control. All drives should normally be listed as robotically controlled (e.g. TLD, ACS, etc.), but they will be converted to “AVR” control if a problem occurs with the robot. There should be an error message in the system logs (/var/adm/messages, etc.) that will help you diagnose the cause of this problem. You can also use the robotic test utilities (e.g. tldtest) to help further debug the problem.

Bpduplicate/Bptm Hanging

A possible problem has been identified where the duplication process itself hangs. This is apparently caused by hardware failures which are logged in system log files (e.g. /var/adm/messages) if using verbose mode with Media Manager daemons:

```
May 5 11:46:51 nirvana acsd[361]: DecodeMount() Actual status:
STATUS_LIBRARY_FAILURE

May 4 03:39:32 inxs unix: SCSI transport failed: reason ‘timeout’: giving up
```

You may need to kill bptm in this case. In either case, you will need to manually determine which bptm is hanging by reviewing “ps” or “bpps” output and looking for older “bptm -dup” and “bptm -copy” commands that are running but not progressing. There is currently no automated way to look for this situation. There are 4 “bptm” processes associated with each bpvault duplication process. One set of “dup” processes and one set of “copy” processes. Each set has a parent and a child. The parent “bptm” has a job parent PID of 1, the child has a parent PID corresponding to the parent “bptm” process. Kill the child bptm first. The parent should automatically end once it unmounts used media. You should see the unmount commands in syslog after you kill the child process.

Killing these jobs manually should allow bpvault to continue. The current image in progress can be duplicated the next day or a second duplicate session can be run.

If bptm doesn’t die after a few minutes, then the error may be related to a SCSI command that won’t complete. In this case, it will be necessary to eventually reboot the machine because the kernel won’t allow bptm to be killed. It may also be necessary to manually complete the current duplication process using bpvault.menu and the steps documented there.

If bptm cannot be killed, you can kill the related bpduplicate. The only way to tell which is the correct bpduplicate is by looking at the start date of the process. Once bpduplicate is killed, bpvault should continue processing other duplicates (unless you have run haltdups.now).

However, some tape(s) may still remain “mounted” in the tape drive(s). You can use the Media Manager GUI to reset the drive. You would need to manually cross-reference which tapes were left mounted by bptm. If using ACSLS, sometimes forcing the tapes to dismount by using the cmd_proc doesn’t seem to effect this situation. In this case, you can let bpvault continue, but you will eventually need to kill “acsd” and “ltid” or reboot the machine to get back in sync. If using TLD controlled robots, you would kill “tldd” and/or “tldcd” along with “ltid”. If this seems confusing, it unfortunately is so because bptm should never hang in the first place.

Note that there is normally no necessity to kill bpvault processes. That is, the bptm process should be killed and allows bpduplicate to complete automatically. Thus bpvault will either continue duplicating more images or exit if it is done or haltdups.now has been run. Once bpvault is finished duplicating, bpvault.all will continue with the session. Once that is finished, you may start bpvault.all again to start a second session for the day to pick up images missed, or you may allow bpvault to duplicate these images the next day.

If bptm doesn’t die, the situation is much more complicated and rebooting the machine at some point is necessary.

“No Duplicate Progress” message

If you begin to see the following message in either “log.file” or “bpvault.all.output_xxx”:

```
bpduplicate[1]: server1[1]:  DUP1: 06/08/98 17:45:14 no duplicate progress message
in 30 minutes
```

Then the bpvault process has not received any new information from the bpduplicate process within the time frame specified (in this case, 30 minutes; this can be changed by using the “no_message_minutes” parameter). This message does not necessarily indicate an error has occurred. If the image that is currently being duplicated is very large, (several gigabytes) then this message is displayed only for informational purposes. To determine if this represents a problem, you may wish to find out the size of the current image. To do this, look at last few lines of the file “bpvault.all.output_xxx” for a line that looks like:

```
./bpvault[1]: server1[1]: starting bpduplicate[1]: server1[1] for image
server2_0897273363 at 06/08/98 17:15:12, count 65 of 88
```

This will tell you the backup image ID. With this information, execute the following command from the shell as root (bpimagelist is located in /usr/openv/netbackup/bin/admincmd):

```
# bpimagelist –L -backupid server2_0897273363
```

The output of this command will show you various statistics about this backup image, including the number of kilobytes written during this backup. If the number is relatively small, then there may be a problem with the duplication process. Sometimes this delay is caused by a media mount (which normally does not occur in robotic devices during duplication) or hardware problems. Examine the Device Management interface to determine if there are any hardware problems and also check the system logs (e.g. /var/adm/messages). If the backup image is very large, then this message should be regarded as informational.

Ejecting Tapes While in Use

If bpvault is configured to eject original media, it is possible that a piece of media could be scheduled for ejection while still being used in a tape drive (e.g. restores, media verify, etc.) If a media is attempted to be ejected from a Media Manager controlled robot, an error message will most likely be generated by Media Manager such as (from bpvault.all.output_xxx):

```
./bpvault: ejecting media ABC123
ABC123:  change failed - Robot busy, cannot perform operation (223)
```

A similar error may be generated by non-Media Manager controlled robots if a media is currently in use. If this occurs, it is recommended that the operator or administrator use the operations menu “bpvault.opsmenu” to re-eject the media after the media is no longer in use. If this method is used, other error messages may be generated stating that some media cannot be ejected since they are not present within the robot. However, the media that generated the previous error should be ejected since it is no longer in use.

Solaris Problems with Port Usage

During field testing, a problem was found with Sun Solaris 2.5 or 2.5.1 and the duplication process. The problem involved running numerous duplication processes that would consume a large number of TCP/IP ports. Due to the default kernel parameters of Solaris, this would cause the operating system to run out of reserved ports and cause NetBackup to fail during duplication. Certain bpduplicate processes would occasionally die with error code 2, 25, 50 or 58. Examples of these error messages (from bpvault.all.output_xxx and log.file) are listed below:

```
./bpvault[1]: server1 [1]: found ERROR: vault: V1, session: DUP1, error line:
09:49:16 INF - bpduplicate terminated by signal (2), other processes may continue
running or terminate ungracefully

./bpvault[1]: server1 [1]: found ERROR: vault: V1, session: DUP1, error line:
08:58:03 INF - host server1  backup id server3_0892110125 read failed, cannot
connect on socket (25).

./bpvault[0]: server1 [0]: found ERROR: vault: V1, session: DUP1, error line:
11:11:05 INF - host server1  backup id server2_0892110120 read process failed,
client process aborted (50).

11:48:54 INF - failure logging message to client server1 in log
/usr/openv/netbackup/vault/V1/DUP1/bpvault.error.0: can't connect to client (58)
```

The issue stems from the operating system leaving TCP/IP ports open and waiting for a response for a certain amount of time before closing the port. The problem has been seen with reserved ports being used up during duplication or other actions which require many reserved ports (e.g. multiplexed backups). According to customer support, the NetBackup logs (in particular, the bptm log) usually shows the message "resource temporarily unavailable" after a call to the procedure getsockconnect().

One fix for this problem that has been given to us by customer support involves modifying the Solaris TCP/IP driver using ndd. On Solaris the ndd command can be used to set driver configuration parameters. The parameter that should be modified is “tcp_close_wait_interval”, and can be modified by executing the following command from the Solaris server as root:

```
# /usr/sbin/ndd -set /dev/tcp tcp_close_wait_interval 1000
```

This parameter can be modified on a running system without rebooting. To retain the value after a reboot the startup process for inetd should be modified by adding the above "ndd" line. This startup process is located in “/etc/init.d/inetinit”. If you have Solaris 2.5 or 2.5.1, you should see an example of an ndd line in this file already.

Another fix for this problem has been identified as a Solaris 2.5/2.5.1 patch which will help clear up the TCP/IP port utilization.

According to SunSolve, patch number 103582 fixes a number of issues with /kernel/drv/tcp. As of this printing, the current revision of this patch is rev. 16 (i.e. 103582-16). It is recommended that if you are running Solaris 2.5 or 2.5.1, you should obtain this patch from Sun. The patch from can be obtained from http://sunsolve.sun.com or by calling your Sun support center. This patch is also included in the Solaris 2.5.1 Patch Cluster, which can be obtained from the above web site in the public patch access area.

VERITAS recommends that all users of bpvault that are using a Sun Solaris master or Sun Solaris slave servers implement this change to avoid problems with running out of TCP/IP ports during the duplication process.

Ejecting More Media Than Export Capacity

If your vault is configured for automatic ejection (parameter “robot_eject” is set to “auto”), you may encounter a situation where more media is scheduled to be ejected than the number of export slots available in your robot. If this happens the pick lists for the robot will show more media than was ejected. To resolve this situation, use the operations menu (bpvault.opsmenu) to eject the tapes for that particular session. For example, if the export capacity of your robot is 14 and bpvault has marked 16 media for ejection, the pick list reports will show 16 tapes while the robot will only be able to eject 14 tapes. Error messages will also be displayed in the log file bpvault.all.output_xxx. The operations menu should be executed and the eject tapes option should be selected. Error messages will appear for the first 14 tapes to be ejected (since they have already been ejected from the robot) while the remaining two tapes will be ejected successfully. The remaining reports (detailed lists, pick lists from vault, etc.) should be reprinted since they depend on media residing in the offsite volume group (see parameter “vault_group”).

If your vault is configured for manual ejection then this problem should not occur in your environment. The result of defining “robot_eject” as “manual” is that a pause statement will be placed in the robot ejection script (eject_tapes for TLD, bpvault.eject for ACSLS) so that an operator will be able to empty the export slots and signal the ejection process to continue. If a vault is configured this way an operator will manually execute the tape ejection process from “bpvault.opsmenu”.

Procedures – Miscellaneous

This section discusses miscellaneous issues that have been raised during field distribution:

Inject Process for TLD Robots

Due to the way in which Media Manager works, it is not trivial to inject tapes into a TLD controlled robot. This is due to the fact that the Media Manager robot inventory functions do not inventory the export slots (also known as CAP slots) of a TLD controlled robot. Therefore, a separate shell script had to be written to provide this functionality. This shell script is located in the file /usr/openv/netbackup/vault/production/bpinject. It takes four command line arguments: Media Manager database host (usually the NetBackup master server), device path to robotic control (e.g. /dev/sg/c0t0d0), robot control host (either the master server if it has a robotic controller, or a slave server), and Media Manager robot number (be sure to specify the correct robot number corresponding to the robot host and device pathname; use the Media Manager GUI or tpconfig –d to verify the robot number).

This script can run as a standalone script on the slave server (provided the directories /usr/openv/netbackup/vault/production and /usr/openv/netbackup/vault/logs exist), or on the master server. If executed on the master server and the robot control host is NOT the master server (i.e. a slave server), then bpinject will attempt to use a remote shell command (rsh) to contact the other server. For this operation to be successful, it is necessary to define the master server in the /.rhosts file on the slave server. You may have also had to modify this script during the installation procedure to define the proper RSH and GREP commands (see the bpinject script for more information).

Once the four command line parameters are established, bpinject will scan the export slots of the robot using tldtest, and determine where to place the tapes in the library. bpinject will also verify that enough empty slots are present in the library before processing the injects. The script will then move the tapes (one at a time) into empty slots in the library, and update Media Manager appropriately. The tapes will be moved into the default volume group for the robot, so you may need to change this to the bpvault volume group (specified as “robot_group” in the parameter file) after the inject process has finished.

As the script is running, output will be directed to the terminal and also a log file in /usr/openv/netbackup/vault/logs/bpinject. After the process is completed, this log file will be renamed to bpinject.<time>, where <time> is the actual time when the process completed.

NOTE: This script assumes a media type of DLT and a robot type of TLD. It also assumes that the barcode of the media is the same as the media ID (which is usually the case). If your configuration differs from this, you may encounter errors when attempting to use this script. In order to allow bpinject to process barcode rules and deal with different media types, it is recommended to use the bpinject-barcode script.

For more details, please read the comments within the bpinject script.

A new feature that has been added to the core Media Manager product is the following command:

```
vmupdate –rn <robot-number> -rt <robot-type> -empty_ie
```

The “-empty_ie” argument will check the contents of the export door and move all media into the library, similar to the bpinject script.

Removing Tapes From Bpvault Control

It may be necessary to remove certain tapes from bpvault due to certain circumstances (e.g. proof of concept testing, failed tape, etc.). The normal procedure for removing a tape from NetBackup

should be followed as per the NetBackup Release Notes and NetBackup System Administrator's Guide by using the command /usr/openv/netbackup/bin/admincmd/bpexpdate. This will cause the media to be unassigned from Media Manager. However, in order to inform bpvault that this media is no longer using a slot at the vault vendor, you must manually remove the "Description" field from the media. This can be accomplished by using the Media Manager GUI, selecting the volume, selecting "Actions" and "Change", selecting the "Description" button, and typing some text in the text box. The text used in this text box is irrelevant, as long as it is not of the bpvault recognized format (e.g. V1|3|S101|00000000); usually we use a single dash character "-". Then click the "OK" button to update Media Manager.

This procedure should be used on media that have already been used for vaulting (e.g. duplicates exist on these tapes and slot numbers have been assigned to them), and for those tapes that wish to be reused with bpvault.

The same objective could be accomplished by expiring the media (using bpexpdate), deleting the volume from the Media Manager GUI (select volume, "Actions", "Delete"), and re-adding the volume (using "Actions", "Add" or using the robot inventory feature).

Command Line Parameters of Bpvault

It should be noted that the bpvault binary expects to be passed certain command line parameters for all functions (media, report, etc.). Each of these are defined below:

Bpvault Session ID: -dupid <session-id>

This argument specifies the session number to use for most of the bpvault commands (e.g. media, report, etc.). It is very important to make sure that this session number is specified correctly since most bpvault commands will depend on the information obtained in the session working directory, which is specified by this parameter. The function "id" and command "new_dup_id" is the only operation that does not require this parameter, since the session ID is created with this operation.

The syntax for "session-id" is an integer. Therefore, if notice certain reports or menus describe the session as "DUP86", then the "session-id" specified here should simply be "86".

Parameter File: -param <dup-param-file>

This argument specifies the file name of the parameter file to be used. A recommended name for this file is "dup_param", but the name can be any valid UNIX file name. The parameters used in this file are documented in the Appendix. If the full pathname to the file is not supplied, this file is expected to reside in the directory in which the bpvault binary is launched, which is usually /usr/openv/netbackup/vault/production.

Preview Mode File: -file <preview-mode-file>

This argument specifies the file that contains the output from the bpvault preview mode (see function "preview" above). It is recommended to use the standard file name "dup.preview.out". This file is expected to reside in the session working directory (e.g. /usr/openv/netbackup/vault/V1/DUP26).

Bpvault Output File: -output <bpvault-output-file>

This argument specifies the file that is used for writing all output from a given bpvault function and command. If debugging mode is used (see parameter "debug"), the output is written to this file. The information in this file is generated internally by the bpvault code. The file is expected to reside in the directory in which the bpvault binary is launched.

Duplicate Progress Log File: -log <command-line-interface-log-file>

This argument specifies the file that is used for writing information regarding the progress of any duplication process. As each backup image is duplicated, a message is written to this file that the image was either duplicated successfully or a failure occurred. See “Procedures – Monitoring” for other discussions of this file. It is recommended to use the file name /usr/openv/netbackup/vault/$VAULT/log.file for this parameter.

Multiple Media Types

It is possible for bpvault to manage multiple media types within the same vault. However, to accomplish this objective it is required to use multiple parameter files. In most cases, duplication would involve writing the second copy of an image to a different media type for offsite storage (e.g. DLT cartridge to ½” cartridge). Therefore, it is recommended to use a parameter file for each unique media type.

For example, if some number of classes should be written to ½” cartridge for offsite storage, while a different set of classes should be written to DLT cartridge, and the original backup images were all written to a single media type (e.g. DLT cartridge), a parameter file would be setup for each set of classes.

First, define the set of classes within each parameter file that should be duplicated. Then define the storage unit that should be used for the destination copy of the image. In this case, one parameter file would list the DLT cartridge storage unit (see parameter “dstunit”), and the other parameter file would list the ½” cartridge storage unit. If both types of drives are attached to the same NetBackup server, then these parameters would be the only differences between the two parameter files. If all of the images within both sets of classes were backed up on the same NetBackup server, the “server” parameter would be the same. This situation would also apply to multiple media densities (e.g. DLT4000 vs. DLT7000, 4890 format vs. Timberline format, etc.) since NetBackup would recognize the media differently (e.g. DLT vs. DLT2, HCART vs. HCART2).

If it is required for the duplicate images to be stored on the same media type as the original, and if the NetBackup server possesses enough resources (i.e. at least two drives of each media type), then separate parameter files may not be necessary. For example, if Server A was connected to two DLT tape drives and two Timberline drives, and some set of backup images were written on both DLT and ½” cartridges, then NetBackup will be able to handle the mount requests and assignment of tape drives automatically within the bpduplicate interface. An example in the first parameter file would be:

```
...
drive_pairs 1
server ServerA 1
dstunit ServerA-DLT 1
...
```

And the second parameter file would look like:

```
...
drive_pairs 1
server ServerA 1
dstunit ServerA-HCART 1
...
```

The classes that were written to DLT media would be placed in the first parameter file, and the classes that were written to ½” cartridges would be placed in the second parameter file.

For more complicated environments, please contact customer support.

Re-ejecting Original Media

If bpvault is configured to eject original media then there may be a time that this original media must be recalled from the vault to process a restore request. After the restore request is complete, the original media should be re-ejected from the robot and transported to the offsite vault.

Currently, there is a manual process that should be followed to re-eject original media after being used for restores. An enhancement request has been logged to automate this process for a future release of the product.

First, determine which session(s) the media was (or were) originally ejected from. You can derive this information by examining the description field for the particular media in the Media Management interface. A command line interface is also available to query the Media Manager database. For example, we have determined that media ID ABC123 was returned from the vault to process a restore request and now must be re-ejected from the robot. The command line interface “vmquery” can be used to determine which session this media was originally ejected from:

```
# /usr/openv/volmgr/bin/vmquery –m ABC123
==========================================================================
media ID:              ABC123
media type:            DLT cartridge tape (11)
barcode:               ABC123
description:           V1|89|S218|00000000
volume pool:           Duplicates (2)
robot type:            TLD – Tape Library DLT (1)
volume group:          vault_grp
created:               Tue Sep 3 10:08:32 1997
assigned:              Tue May 6 00:11:45 1998
last mounted:          Tue May 6 11:34:25 1998
first mount:           Tue Sep 3 18:20:48 1997
expiration date:       ---
number of mounts:      21
max mounts allowed:    ---
status:                0x0
==========================================================================
```

From the description field we can see that this media was ejected during session 89 from vault “V1” and belongs in offsite slot 218.

Next, determine which type of robotics are being used and choose the appropriate eject method below:

For Media Manager controlled robots, execute the vmchange command as follows:

```
/usr/openv/volmgr/bin/vmchange -vh <master-server> -res -m <media> -mt <media-
       type> -rt none -v <vault-group> -rc1 0 -rc2 0 -e -sec 5
```

Where:

```
<master-server> is the host name of the Media Manager database host, which is
       usually the master server
<media> is the media ID to be ejected (e.g. ABC123)
<media-type> is the media type (e.g. DLT, 8MM, etc.)
<vault-group> is the volume group name that is defined as “vault_group” in the
       parameter file for this vault
```

This will eject the media from the robot it is currently attached to and move the media to the appropriate volume group.

For ACSLS robots, execute the following vmchange commands as follows:

```
/usr/openv/volmgr/bin/vmchange -vh <master-server> -new_rt none -m <media>
/usr/openv/volmgr/bin/vmchange -vh <master-server> -new_v <vault-group> -m <media>
```

Where:

```
<master-server> is the host name of the Media Manager database host, which is
        usually the master server
<media> is the media ID to be ejected (e.g. ABC123)
<vault-group> is the volume group name that is defined as “vault_group” in the
        parameter file for this vault
```

This will update the Media Manager database to show the media in a non-robotic state. Then, you must access the ACSLS console (either locally or by telnet to the host and executing cmd_proc as the user acsss) to physically eject the media with the commands:

```
eject <cap-location> <media>
logoff
```

Where:

```
<cap-location> is the CAP identifier (e.g. 0,0,0)
<media> is the media ID to be ejected (e.g. ABC123)
```

For more information on ACSLS commands, see the StorageTek ACSLS command reference.

Allowing Non-Root Access for Operations

In order to allow non-root users access to operational commands (those listed in bpvault.opsmenu), the following steps must be followed.

1. Create a UNIX group for the operations staff and make sure all UNIX accounts belong to this group. This group will be referred to as “<opsgroup>” in the commands below.
2. Log in to the NetBackup master server and execute the following commands as root:

   ```
   # cd /usr/openv/netbackup/vault
   # chmod -R 775 <vault> logs production production/bpvault.*
   # chgrp -R <opsgroup> <vault> logs production production/bpvault.*
   # chmod +s <vault> logs
   # cd /usr/openv/volmgr/bin
   # chmod 4550 tldtest vmchange vmquery vmadd vmupdate
   # chgrp <opsgroup> tldtest vmchange vmquery vmadd vmupdate
   ```

   Where:

   ```
   <vault> is/are the name(s) of the vault directory/directories.
   ```

3. Modify the file “/usr/openv/netbackup/vault/production/bpvault.env” and uncomment the following (at about line 86):

   ```
   umask 002
   ```

4. Log in to any NetBackup server that is controlling a TLD robot and execute the following commands as root:

   ```
   # cd /usr/openv/volmgr/bin
   # chmod 4550 tldtest vmchange vmquery vmadd vmupdate
   # chgrp <opsgroup> tldtest vmchange vmquery vmadd vmupdate
   ```

5. If using ACSLS for robotic control, log in to the ACSLS server and modify the .rhosts file in the home directory of user “acsss”. There should already be an entry for the master server (added during the install process) granting root access. The entry should be modified to allow all users in the operations group access to the ACSLS server as user “acsss”:

   ```
   master-server root <user> ... <user>
   ```

6. If using ACSLS for robotic control, verify that the user accounts defined for the operations group exist on both the master server and the ACSLS server. For simplicity, it is recommended to assign the same user ID on both servers for each user. If both servers are using a naming service (e.g. NIS, NIS+), then this step will most likely be skipped.

To verify proper operation, log in to the master server as an operations user, execute the bpvault.opsmenu and execute each menu option. For more details regarding bpvault.opsmenu, see the Operations Guide.

Appendix A – File and Directory Structure

Bpvault Programs and Scripts	Purpose
/usr/openv/netbackup/vault	All programs, working directories, etc are under this directory.
../production/bpinject	Script for TLD controlled robots used to automatically move tapes from export slots.
../production/bpinject-tlm	Script for TLM controlled robots used to automatically move tapes from export slots.
../production/bpvault	This is a symbolic link (created during installation) to the bpvault binary corresponding to the platform of the master server, which is one of: • bpvault.solaris • bpvault.hp800 • bpvault.rs6000 • bpvault.ncr • bpvault.sgi • bpvault.sequent • bpvault.dec
../production/bpvault.acs	Script called by bpvault.all after duplication to execute commands structured for ACSLS robots.
../production/bpvault.all	Main shell script customizable for customer's vault setup.
../production/bpvault.all.output_$VAULT	Contains last bpvault.all log file. File is copied to working session subdirectory upon completion.
../production/bpvault.change	Menu to be used for changing bpvault information for specific media.
../production/bpvault.env	Shell script which should be customized for customer's network configuration (e.g. network printer names, email addresses, etc.); this script is executed at the beginning of all other bpvault.* shell scripts to establish various environment variables.
../production/bpvault.menu	Storage Administrator menu; see section entitled "Procedures – bpvault.menu" for more details.
../production/bpvault.opsmenu	Operations menu; see section entitled "Procedures" in Operations Guide for more details.
../production/bpvault.reports	Script called by robot-specific script after ejects to generate reports and send via email or printer or place into repository.
../production/bpvault.tl8	Script called by bpvault.all after duplication to execute commands structured for TL8 robots.
../production/bpvault.tl8.eject	Script called by bpvault.tl8 to physically eject media from TL8 robots.
../production/bpvault.tld	Script called by bpvault.all after duplication to execute commands structured for TLD robots.
../production/bpvault.tld.eject	Script called by bpvault.tld to physically eject media from TLD robots.
../production/bpvault.tlh	Script called by bpvault.all after duplication to execute commands structured for TLH robots.
../production/bpvault.tlh.eject	Script called by bpvault.tlh to physically eject media from TLH robots.

../production/bpvault.tlm	Script called by bpvault.all after duplication to execute commands structured for TLM robots.
../production/bpvault.tlm.eject	Script called by bpvault.tlm to physically eject media from TLM robots.
../production/changecopy.sh	Example script to automate modification of primary copy for disaster recovery purposes.
../production/dup_param.acs_example ../production/dup_param.tl8_example ../production/dup_param.tld_example ../production/dup_param.tlh_example ../production/dup_param.tlm_example	Sample parameter files; should be used as templates for customer specific parameter files.
../production/haltdups.now	Script to signal to bpvault to finish after current duplicate is complete.
../production/version	NetBackup Vault Extension version information
../bin/robot_inventory	Shell script to run robot inventory; **NOTE: should be customized with appropriate hostnames, robot numbers, etc.**
../bin/eject_tapes	Shell script to forward tape ejects to ACSLS.
../<VAULT>/ (e.g. ../CH_V1/) (e.g. ../LSM10)	Subdirectory for working session directories and log file.
../<VAULT>/DUPnnn (e.g. ../CH_V1/DUPnnn) (e.g. ../LSM10/DUPnnn)	Working session subdirectories. These can be manually removed by the customer if necessary to reduce disk usage.
../<VAULT>/log.file (e.g. ../CH_V1/log.file) (e.g. ../LSM10/log.file)	Normal location for image “ok” or “error” message for this vault; should be monitored by administrator or third-party software.
../<VAULT>/bpvault.duplicate	Counter to show current duplication session
../<VAULT>/bpvault.pid	Process id of current bpvault duplication session. Used by haltdups.now to signal soft shutdown.
../<VAULT>/bpvault.times	Success or Error for bpvault duplication session
../<VAULT>/DUPnnn/bpvault.dup.nn	Subset of dup.preview.out. Has images to be duplicated on one of the drive pairs. Created by bpvault as first step in duplicate mode. One file for each drive pair.
../<VAULT>/DUPnnn/bpvault.eject	For ACSLS environment: eject command log file; created by bpvault “eject” mode from running ../bin/eject_tapes script.
../<VAULT>/DUPnnn/bpvault.eject.log	For ACSLS environment: log file with results of executing ../bin/eject_tapes script using bpvault.eject as input.
../<VAULT>/DUPnnn/bpvault.error.file	Error log for other administrative commands performed by bpvault (e.g. preview mode, modifying Media Manager database); this file should be checked in case of problems.
../<VAULT>/DUPnnn/bpvault.error.file_suspend	Error log for other administrative commands performed by bpvault during suspend mode; this file should be checked in case of problems.
../<VAULT>/DUPnnn/bpvault.error.nn	Error log for duplication of specific image.
../<VAULT>/DUPnnn/bpvault.log.nn	Log file for duplication of image. Created by bpduplicate and contains output from bptm. One file for each drive pair.
../<VAULT>/DUPnnn/bpvault.prev.file	Output file that is monitored during preview mode to determine when preview mode is complete.
../<VAULT>/DUPnnn/bpvault.prev.file_susp	Output file that is monitored during suspend mode

end	to determine when preview routine is complete.
../<VAULT>/DUPnnn/class.inventory	Listing of all configured NetBackup classes; for use with recovery report.
../<VAULT>/DUPnnn/dup.preview.out	Duplication preview output file. Shows images which are to be duplicated. Created by bpvault during preview mode.
../<VAULT>/DUPnnn/dup.preview.out_suspend	List of images for which media will be suspended. Created by bpvault during suspend mode.
../<VAULT>/DUPnnn/eject_tapes	For TLD environment: Shell script with vmchange commands that will eject tapes for this session and change tapes to non-robotic vault group.
../<VAULT>/DUPnnn/image.inventory	NetBackup image catalog information for each image duplicated. Created by bpvault “image_inv” mode (which uses bpimagelist output).
../<VAULT>/DUPnnn/image.inventory_suspend	NetBackup image catalog information for each image whose media will be suspended. Created by bpvault “image_inv” portion of suspend mode (which uses bpimagelist output).
../<VAULT>/DUPnnn/media.inventory	NetBackup media used for originals and duplicates. Created by bpvault “media_inv” mode (which uses data in image.inventory).
../<VAULT>/DUPnnn/media.inventory_suspend	NetBackup media used for originals and to be suspended. Created by bpvault “media_inv” portion of suspend mode (which uses data in image.inventory).
../<VAULT>/DUPnnn/nb_media.inventory	Contains # of images, size of images, expiration dates on original and duplicated media. Used for reports. Created by bpvault “media_inv” mode (which uses bpmedialist output).
../<VAULT>/DUPnnn/nb_media.inventory_suspend	Contains # of images, size of images, expiration dates on original media to be suspended. Used for reports. Created by bpvault “media_inv” portion of suspend mode (which uses bpmedialist output).
../<VAULT>/DUPnnn/nbudb.media	Contains media IDs that were allocated for NBU database backups. Created by bpvault “assigndb_dup” mode (which uses vmquery output to select an appropriate media).
../<VAULT>/DUPnnn/rcvrimage.inventory	NetBackup image catalog information for all classes between dates specified in parameter file (recoverto_days, recoverlength_days); for use with recovery report.
../<VAULT>/DUPnnn/robot.inventory	File generated by ./bin/robot_inventory script that lists all media currently residing in robot (one media ID per line).
../<VAULT>/DUPnnn/scratch.inventory	Temporary file used for expiring recalled tapes from vault.
../<VAULT>/DUPnnn/volume.inventory	Media Manager inventory for duplicate pool (parameter(s) “pool”) and NBU database duplicate pool (parameter “dbpool”). Created by bpvault “volume_inv” mode (which uses vmquery output).
../<VAULT>/DUPnnn/volume.inventory_suspend	Media Manager inventory for original media pool(s) (parameter(s) “pool”). Created by bpvault “volume_inv” portion of suspend mode (which uses vmquery output).
../<VAULT>/DUPnnn/volume.inventory.db	Media Manager inventory for NBU database

	duplicate pool (parameter “dbpool”). Created by bpvault “volume_inv” mode (which uses vmquery output).
../<VAULT>/DUPnnn/volume.inventory.dup	Media Manager inventory for duplicate pool (parameter(s) “pool”). Created by bpvault “volume_inv” mode (which uses vmquery output).
../<VAULT>/DUPnnn/volume.inventory.dup_suspend	Media Manager inventory for original media pool(s) (parameter(s) “pool”). Created by bpvault “volume_inv” portion of suspend mode (which uses vmquery output).
../<VAULT>/DUPnnn/volume_full.inventory	Media Manager inventory of all media. Created by bpvault “volume_inv_full” mode (which uses vmquery output).
/usr/openv/netbackup/logs/admin	Log file for bpduplicate
/usr/openv/netbackup/logs/bptm	Log file for tape manager
/usr/openv/netbackup/logs/bpdbm	Log file for database queries
/usr/openv/netbackup/bp.conf	Set VERBOSE option here for NetBackup logs
/usr/openv/netbackup/db/media/tpreq	Shows list of currently requested media on server

Appendix B – Parameter File

The parameter file for bpvault accepts the following parameters, listed in alphabetical order.

NOTE: An asterisk (*) denotes parameters that MUST be defined within the parameter file.

NOTE: Any lines within the parameter file that begin with a pound sign (#) will be ignored and assumed to be comments.

Parameter	Description	Values	Default	Example
acs	ACS number to use for duplication	Integer	0	acs 0
acs_ack_timeout	Number of seconds to wait for an acknowledgement response from the ACSLM	Integer	30	acs_ack_timeout 60
acs_cap_0	First CAP number defined for the ACS and LSM used	Integer	0	acs_cap_0 0
acs_cap_1	Second CAP number defined for the ACS and LSM used	Integer	1	acs_cap_1 1
acs_cap_count	Number of slots available in the LSM CAPs	Integer	40	acs_cap_count 40
acs_cap_num	Number of CAPs defined in the ACS	Integer	2	acs_cap_count 1
acs_command_timeout	Number of seconds to wait for a command response from the ACSLM	Integer	300	acs_command_timeout 600
acs_csi_hostname	Hostname of server that controls ACSLM; used for communication with ACS API	ASCII	- NONE -	acs_csi_hostname acsls-server
acs_lockfile_prefix	Name of acsssi lockfile used by Media Manager. It is not recommended to change this parameter.	ASCII	acsssi.lock.	acs_lockfile_prefix acsssi.lock.
acs_lsm	LSM number to be used within the defined ACS	Integer	1	acs_lsm 0

Parameter	Description	Values	Default	Example
acs_nodrives_onmaster	Used when communicating with ACS API. If no tape drives are defined on the master server, set this to “1”. Otherwise, set to “0”.	Integer	0	acs_nodrives_onmaster 1
acs_number_tries	Number of times to wait for a "command complete" response from the ACSLM	Integer	5	acs_number_tries 10
acs_sockname	Socket number to use for communication with acsssi process to eject tapes. This parameter should not need to be modified unless the default port is already used.	ASCII	13771	acs_sockname 13781
assignslot_multvault	During slot assignment, only look at volumes within a given vault for the next unique slot number; used for sites with a single volume pool spanning multiple robots.			assignslot_multvault 1
backupdb_command	Name and location of NetBackup database backup command	ASCII	/usr/openv/netbackup/bin/admincmd/bpbackupdb	backupdb_command /usr/openv/netbackup/bin/admincmd/bpbackupdb
bpexpdate_command	Name and location of the bpexpdate command	ASCII	/usr/openv/netbackup/bin/admincmd/bpexpdate	bpmedia_command /usr/openv/netbackup/bin/admincmd/bpexpdate
bpmedia_command	Name and location of the bpmedia command	ASCII	/usr/openv/netbackup/bin/admincmd/bpmedia	bpmedia_command /usr/openv/netbackup/bin/admincmd/bpmedia
bpmedia_option	Option to be used with freezing or suspending of media; freeze - set media to freeze state which allows no further writing, and tape will never expire OR suspend - set media to suspend state which allows no further writing, and tape	ASCII	suspend	bpmedia_option suspend

Parameter	Description	Values	Default	Example
	will expire normally			
bpvault_dir *	Working directory for this vault.	ASCII	- NONE -	bpvault_dir /usr/openv/netbackup/vault/CH_V1
changeexpdate	Toggle used to enable changing of expiration dates for duplicate images. Set this value to “1” to enable feature.	Integer, 0 or 1	0	changeexpdate 1
change_command	Name and location of NetBackup media change command	ASCII	/usr/openv/volmgr /bin/vmchange	change_command /usr/openv/volmgr/bin/vmchange
class *	One line for each class to be duplicated; **NOTE: use the parameter “all” to vault all classes**	ASCII	- NONE -	class apollo_BR1 (or) class all
class_sort_field	Field number to sort on when generating preview mode output; it is not recommended to change this parameter.	Integer	2	class_sort_field 2
classexclude	If class parameter is defined as “all”, this parameter defines classes that should NOT be vaulted; one line for each classexclude	ASCII	- NONE -	classexclude test_class
cllist_command	Name and location of NetBackup class list command.	ASCII	/usr/openv/netbackup/bin/admincmd/bpcllist	cllist_command /usr/openv/netbackup/bin/admincmd/bpcllist
command_log	Used to monitor NetBackup daemons on the master server. Bpvault will place this log into the output file if the log directory exists.	ASCII	- NONE -	command_log /usr/openv/netbackup/logs/bptm command_log /usr/openv/netbackup/logs/bpdbm command_log /usr/openv/netbackup/logs/admin
compinv_header	Header to be used for Complete Inventory List report; command line report name “inventory”	ASCII	Complete Inventory List	compinv_header Complete Inventory List
copy_number	Copy number of image to use as	Integer, 1 or 2	1	copy_number 1

Parameter	Description	Values	Default	Example
	source for duplication; this should normally be set to 1.			
dasadmin_client	Host on which the “dasadmin” utility will be called	ASCII	<master server; determined at runtime using gethostname()>	dasadmin_client slaveserver
dasadmin_command	Location of “dasadmin” command; it is not recommended to change this parameter, unless the DAS software is loaded in a different directory.	ASCII	/usr/local/aci/dasadmin	dasadmin_command /opt/local/aci/dasadmin
dasadmin_eject	Name of the GRAU eject area.	ASCII	- NONE -	dasadmin_eject 101
dasadmin_eject_complete	Whether to eject completely and free up the library slot (default behavior; value of 1), or to keep slot reserved for reinjection (value of 0).	Integer, 0 or 1	1	dasadmin_eject_complete 1
dasadmin_insert	Name of the GRAU insert area.	ASCII	- NONE -	dasadmin_insert 100
dasadmin_server	Host that physically controls the GRAU library; usually an OS/2 machine.	ASCII	- NONE -	dasadmin_server os2client
days_holddbtape	Number of days after today to hold the vaulted copy of the NBU DB tape.	Integer	30	days_holddbtape 60
days_startfrom	Number of days ago before today to start searching for images. This number should be LESS THAN “duplicate_days”.	Integer	0	days_startfrom 2
days_suspend	Number of days ago before today that you wish to freeze/suspend media. This number should be LESS THAN “days_startfrom”.	Integer	0	days_suspend 1
dbbackuphost	Hostname of server that will backup NetBackup databases for vaulting; master server is default, but should be defined if master server does not	ASCII	<master server; determined at runtime using gethostname()>	dbbackuphost slaveserver

Parameter	Description	Values	Default	Example
	have any tape drives attached.			
dbpath	If defined, use these paths (one per line) for NBU database backup duplicates. If not defined, use paths defined in NetBackup GUI.	ASCII	- NONE -	dbpath masterserver:/usr/openv/netbackup/db dbpath masterserver:/usr/openv/volmgr/database
dbpool *	Name of pool used for NBU database backup duplicates. **NOTE: this pool should NOT be the same as any pool(s) listed under the "pool" parameter.**	ASCII	- NONE -	dbpool NBDB_Duplicate
debug	Set to 1 to get debugged output	Integer	0	debug 0
debug_counter	Number of rows to print out for some debugging information	Integer	10	debug 20
distdtl_header	Header to be used for Detailed Distribution List for Vault report; command line report name "eject_detail"	ASCII	Detailed Dist. List for Vault	distdtl_header Detailed Dist. List for Vault
distlib_header	Header to be used for Distribution List for Library report; command line report name "dist_library"	ASCII	Distribution List for Library	distlib_header Distribution List for Library
distsum_header	Header to be used for Summary Distribution List for Vault report; command line report name "eject_detail_sum"	ASCII	Summary Dist. List for Vault	distsum_header Summary Dist. List for Vault
distvlt_header	Header to be used for Distribution List for Vault report; command line report name "dist_vault"	ASCII	Distribution List for Vault	distvlt_header Distribution List for Vault
drive_pairs *	Set to the total number of pairs configured on all the servers	Integer	0	drive_pairs 8
dstunit	For each server, identify the name of the destination storage unit and the number of drive pairs available through it for duplication, default 1. These are the storage units for	ASCII, Integer	- NONE -, 1	dstunit b_nirvana 1 dstunit d_nirvana 1

Parameter	Description	Values	Default	Example
	creating the duplicated tape - the source tapes should be on the same machine for network efficiency, but can be a different unit and media type.			
dup_cleanup	Used within the function “duplicate” to assign any images that did not match a server name found in the dup_param file. Normally used for allowing network duplication.	0 (off) or 1 (on)	0	dup_cleanup 0
dup_file_template	Prefix used for file which stores images to be duplicated on a given pair of drives; it is not recommended to change this parameter.	ASCII	bpvault.dup.	dup_file_template bpvault.dup.
duplicate_command	Location of bpduplicate command; it is not recommended to change this parameter.	ASCII	/usr/openv/netbackup/bin/admincmd/bpduplicate -v	duplicate_command /usr/openv/netbackup/bin/admincmd/bpduplicate -v
duplicate_days	Number of days to search back for duplicates. Once a duplicate is made, it is skipped over on subsequent bpvault runs.	Integer	30	duplicate_days 7
duplicate_hours	Number of hours ADDED TO "duplicate_days", which sets the START DATE of the preview mode	Integer	0	duplicate_hours 2
eject_command	For ACS robots only: command used to eject tapes from robot. Currently expects name of log file and receives ACSLS “eject 0,1,0 …” type input.	ASCII	/usr/openv/netbackup/vault/bin/eject_tapes	eject_command /usr/openv/netbackup/vault/bin/eject_tapes
eject_command_log	For ACS robots only: file to store output of tape ejection commands; it is not recommended to change this parameter.	ASCII	bpvault.eject.log	eject_command_log bpvault.eject.log

Parameter	Description	Values	Default	Example
eject_header	Header to print on all reports	ASCII	- NONE -	eject_header Wells Fargo - Cassie Hills
eject_repeat	For ACS robots; issue the same tape eject commands this number of times. Normally used for debugging purposes.	Integer (>= 1)	1	eject_repeat 1
error_file_template	Prefix used for file which stores output from bpduplicate. Used by bpvault to determine if image was successfully duplicated. It is not recommended to change this parameter.	ASCII	bpvault.error.	error_file_template bpvault.error.
fullvlt_header	Header to be used for Full Inventory List for Vault report; command line report name “vault_full”	ASCII	Full Inventory List for Vault	fullvlt_header Full Inventory List for Vault
hours_startfrom	Number of hours ADDED TO "days_startfrom", which set the END DATE of the preview mode	Integer	0	hours_startfrom 2
imlist_command	Name and location of NetBackup image list command.	ASCII	/usr/openv/netbackup/bin/admincmd/bpimagelist	imlist_command /usr/openv/netbackup/bin/admincmd/bpimagelist
invtvlt_header	Header to be used for Inventory List for Vault report; command line report name “vault”	ASCII	Inventory List for Vault	invtvlt_header Inventory List for Vault
line_max	Number of lines to print on a report before printing the next page header	Integer	58	line_max 55
log_file_template	Prefix used for file which stores any command line errors reported by bpduplicate. It is not recommended to change this parameter.	ASCII	bpvault.log.	log_file_template bpvault.log.
media_type	Type of media used in GRAU library; required for dasadmin commands.	ASCII	- NONE -	media_type 3480
medlist_command	Name and location of NetBackup	ASCII	/usr/openv/netbac	medlist_command

Parameter	Description	Values	Default	Example
	media list command.		kup/bin/admincmd/bpmedialist	/usr/openv/netbackup/bin/admincmd/bpmedialist
mmhost	Hostname of Media Manager database server; this is usually the master server so it does not need to be defined.	ASCII	<master server; determined at runtime using gethostname()>	mmhost slaveserver
mpxdup	Toggle to be used for multiplexed duplication of images; set to “1” to enable multiplexed duplication or set to “0” to enable de-multiplexed duplication.	Integer, 0 or 1	0	mpxdup 1
mtlib_bulk	For TLH robots only: if the IBM3494 has a "bulk export" area set to “1”; set to “0” if not.	Integer	1	mtlib_bulk 0
mtlib_command	For TLH robots only: fully qualified pathname of the "mtlib" command on the host that controls the TLH robot.	ASCII	/usr/bin/mtlib	mtlib_command /usr/bin/mtlib
mtlib_device	For TLH robots only: device file which corresponds to the TLH robot; should exist on the host which is controlling the TLH robot and can be obtained from "tpconfig -d" or "xdevadm"	ASCII	/dev/lmcp0	mtlib_device /dev/lmcp1
mtlib_host	For TLH robots only: host which controls the robot; not necessary to define this parameter if the robot is being controlled by the master server.	ASCII	<master server; determined at runtime using gethostname()>	mtlib_host robothost
multrobots_onserver	Toggle to be used if there are multiple robots attached to a single server; set to “1” if there are multiple robots attached to the server, otherwise set to “0”	Integer, 0 or 1	0	multrobots_onserver 1

Parameter	Description	Values	Default	Example
multrobots_onserver_srcgrp	Used in conjunction with “multrobots_onserver”; used to define the volume group where the SOURCE images reside	ASCII	<same as “robot_group”>	multrobots_onserver_srcgrp 00_001_TLD
nb_version	Version of NetBackup running on master server	Integer (2 or 3)	3	nb_version 3
no_message_minutes	Number of minutes during bpduplicate to wait before reporting possible hung condition	Integer	30	no_message_minutes 30
num_dbdups	Number of NetBackup database backups to perform to unique media; all of these copies will be ejected and sent offsite. Define this parameter as 0 to disable offsite NetBackup database backup.	Integer	0	num_dbdups 2
origdtl_header	Header to be used for Detailed Original Media Eject List report; command line report name “eject_orig_detail”	ASCII	Detailed Original Media Eject List	origdtl_header Detailed Original Media Eject List
origejc_header	Header to be used for Original Media Eject List report; command line report name “eject_orig”	ASCII	Original Media Eject List	origejc_header Original Media Eject List
picklib_header	Header to be used for Picking List for Library report; command line report name “picking_library”	ASCII	Picking List for Library	picklib_header Picking List for Library
pickvlt_header	Header to be used for Picking List for Vault report; command line report name “picking_vault”	ASCII	Picking List for Vault	pickvlt_header Picking List for Vault
pool *	Name of pool to use for duplication; multiple pools, one per line, may be defined if vaulting originals. **NOTE: any pool(s) defined with this parameter should NOT also be defined under the parameter**	ASCII	- NONE -	pool Duplicates

Parameter	Description	Values	Default	Example
	“dbpool”.			
print_log_flag	Can turn off printing of bpduplicate log yet will still monitor file. Not recommended.	0 (off) or 1 (on)	1	print_log_flag 1
query_command	Name and location of NetBackup media manager query command	ASCII	/usr/openv/volmgr /bin/vmquery	query_command /usr/openv/volmgr/bin/vmquery
recoverlength_days	Number of days from “recoverto_days” to include in disaster recovery report; see reports for more information	Integer	14	recoverlength_days 30
recoverto_days	Number of days before today to search NBU DB for media necessary to recover all classes and display on disaster recovery report; see reports for more information	Integer	0	recoverto_days 7
recovry_header	Header to be used for Recovery Report for Vault report; command line report name “recovery”	ASCII	Recovery Report for Vault	recovry_header Recovery Report for Vault
remote_command	Name and location of operating system remote (unrestricted) shell command.	ASCII	/usr/bin/rsh	remote_command /usr/bin/remsh
report_dir	Pathname to directory that all reports will be sent to during vault session,	ASCII	/usr/openv/netbackup/vault/reports	report_dir /usr/local/vault/reports
retention_length	Number of days to add to a duplicate image’s expiration date based on the retention level of the image; only used during a duplication vault session. First number defines the retention level, second number defines the number of days to add to the expiration date of duplicated images at that	Integer (0 thru 9), Integer	0 for each retention level	retention_length 5 21

Parameter	Description	Values	Default	Example
	retention level. **NOTE: changeexpdate must be set to "1" to enable this feature.**			
robot_eject	Either wait or do not wait for operator input after library CAP full	auto (do not wait), manual (wait)	auto	robot_eject manual
robot_group	Name of robotic volume group for duplication pool	ASCII	- NONE -	robot_group robot_grp
robot_inventory	Command used to run robot inventory. Must return single column list of media in robot.	ASCII	/usr/openv/netbackup/vault/bin/robot_inventory	robot_inventory /usr/openv/netbackup/vault/bin/robot_inventory
robot_num	Robot identifier to be used for vaulting; this should be the robot number as defined in Media Manager	Integer	0	robot_num 1
robot_type	Type of robot to be used for vaulting	acs, tld	acs	robot_type tld
run_report	Flag to determine if reports should be automatically printed after each vault session.	0 (do not generate reports), 1 (generate reports)	1	run_report 0
run_report_file	Toggle that defines if reports should be sent to the file repository; set to "1" to send reports to repository, otherwise set to "0".	Integer, 0 or 1	1	run_report_file 0
run_report_mail	Toggle that defines if reports should sent via email; set to "1" to send reports via email, otherwise set to "0". **NOTE: MAILPRG and MAILNAMES must be defined in bpvault.env.**	Integer, 0 or 1	1	run_report_mail 0
run_report_print	Toggle that defines if reports should be sent to printer; set to "1" to send reports to printer, otherwise set to	Integer, 0 or 1	1	run_report_print 0

Parameter	Description	Values	Default	Example
	"0". **NOTE: LPR must be defined in bpvault.env.**			
schedule *	Schedule name(s) to search for in preview mode for vaulting, one per line. **NOTE: use schedule "none" to search for all schedules within a given class.**	ASCII	- NONE -	schedule fullbkups OR schedule none
server *	For each server, whether master or slave, enter the name of the server and the number of drive pairs used for duplication	ASCII, Integer	- NONE -, 1	server nirvana 3
sleep	Amount of time to wait before re-reading log file.	Integer	1	sleep 60
starting_slot	Number to begin offsite slot assignment at.	Integer	1	starting_slot 100
suspendmedia	Toggle that defines if media should be suspended after vault session completes. Set to "1" to enable feature, otherwise set to "0". **NOTE: Only available when vaulting original media.**	Integer, 0 or 1	0	suspendmedia 1
tl8_count	Number of available export slots in TL8 controlled robot	Integer	1	tl8_count 2
tl8_eject	Type of eject to perform on TL8 robots; either vmchange (Media Manager controlled) or tl8test (low-level CLI)	ASCII, vmchange or tl8test	vmchange	tl8_eject tl8test
tl8test_bus	For NT TL8 ejects only; bus number of the TL8 robot controller	Integer	0	tl8test_bus 0
tl8test_command	For tl8test ejects only; location of tl8test binary	ASCII	/usr/openv/volmgr /bin/tl8test	tl8test_command /usr/openv/volmgr/bin/tl8test
tl8test_host	For UNIX TL8 ejects only; name of server that controls TL8 robot	ASCII	- NONE -	tl8test_host server1

Parameter	Description	Values	Default	Example
tl8test_lun	For NT TL8 ejects only; LUN number of the TL8 robot controller	Integer	0	tl8test_lun 0
tl8test_port	For NT TL8 ejects only; port number of the TL8 robot controller	Integer	0	tl8test_port 0
tl8test_robotpath	For UNIX TL8 ejects only; device file for TL8 robot control	ASCII	- ASCII -	tl8test_robotpath /deg/sg/c4t2l0
tl8test_target	For NT TL8 ejects only; target number of the TL8 robot controller	Integer	0	tl8test_target 0
tld_count	Number of available export slots in TLD controlled robot	Integer	14	tld_count 4
tlh_count	Number of available export slots in TLH controlled robot	Integer	10	tlh_count 1
tlm_count	Number of available export slots in TLM controlled robot	Integer	10	tlm_count 20
tpreq_command	Name and location of Media Manager tape request command.	ASCII	/usr/openv/volmgr /bin/tpreq	tpreq_command /usr/openv/volmgr/bin/tpreq
tpunmount_command	Name and location of Media Manager tape unmount command.	ASCII	/usr/openv/volmgr /bin/tpunmount	tpunmount_command /usr/openv/volmgr/bin/tpunmount
vault *	Provide a short name for this vault; **NOTE: must be five (5) characters or less**	ASCII	- NONE -	vault CH_V1
vault_group	Name of non-robotic volume group for offsite volumes; **NOTE: must be ten (10) characters or less**	ASCII	- NONE -	vault_group vault_grp
vault_type	Which images should be vaulted: either create duplicates to be vaulted or vault the original images	normal (duplicates), original	normal	vault_type normal
vault_vendor	Name of offsite vendor to be printed on all reports	ASCII	ARCUS	vault_vendor DataBase
vmchange_timeout	Number of seconds that vmchange will wait before robot automatically re-insert the tape into library; usually used on smaller robots (e.g. TL8).	Integer	5	vmchange_timeout 30

Parameter	Description	Values	Default	Example
volmgr_miscdir	Only used for ejects on ACS robots (using ACS API, robot_type acsla) on UNIX master servers; defines location of Media Manager “misc” directory which consists of acsssi lock files.	ASCII	/usr/openv/volmgr /misc	volmgr_miscdir /usr/openv/volmgr/misc

Appendix C – Sample Configurations & Parameter Files

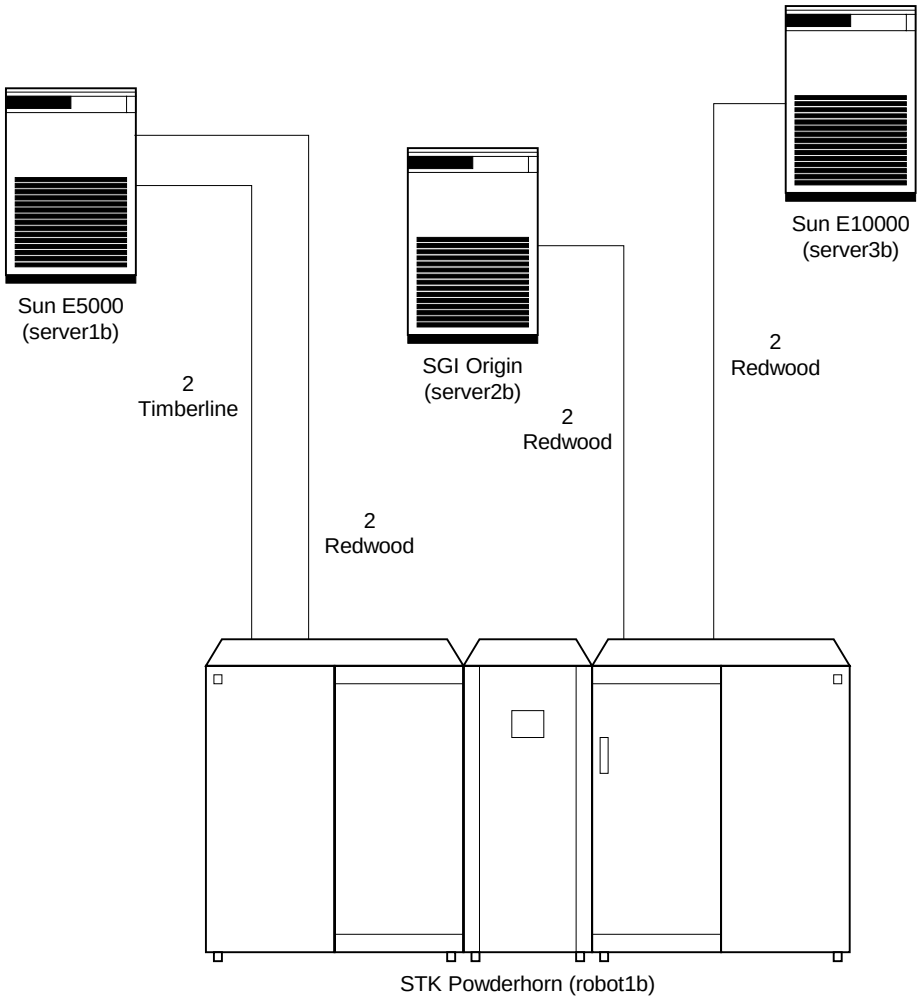


Figure 1. Example ACSLS Configuration

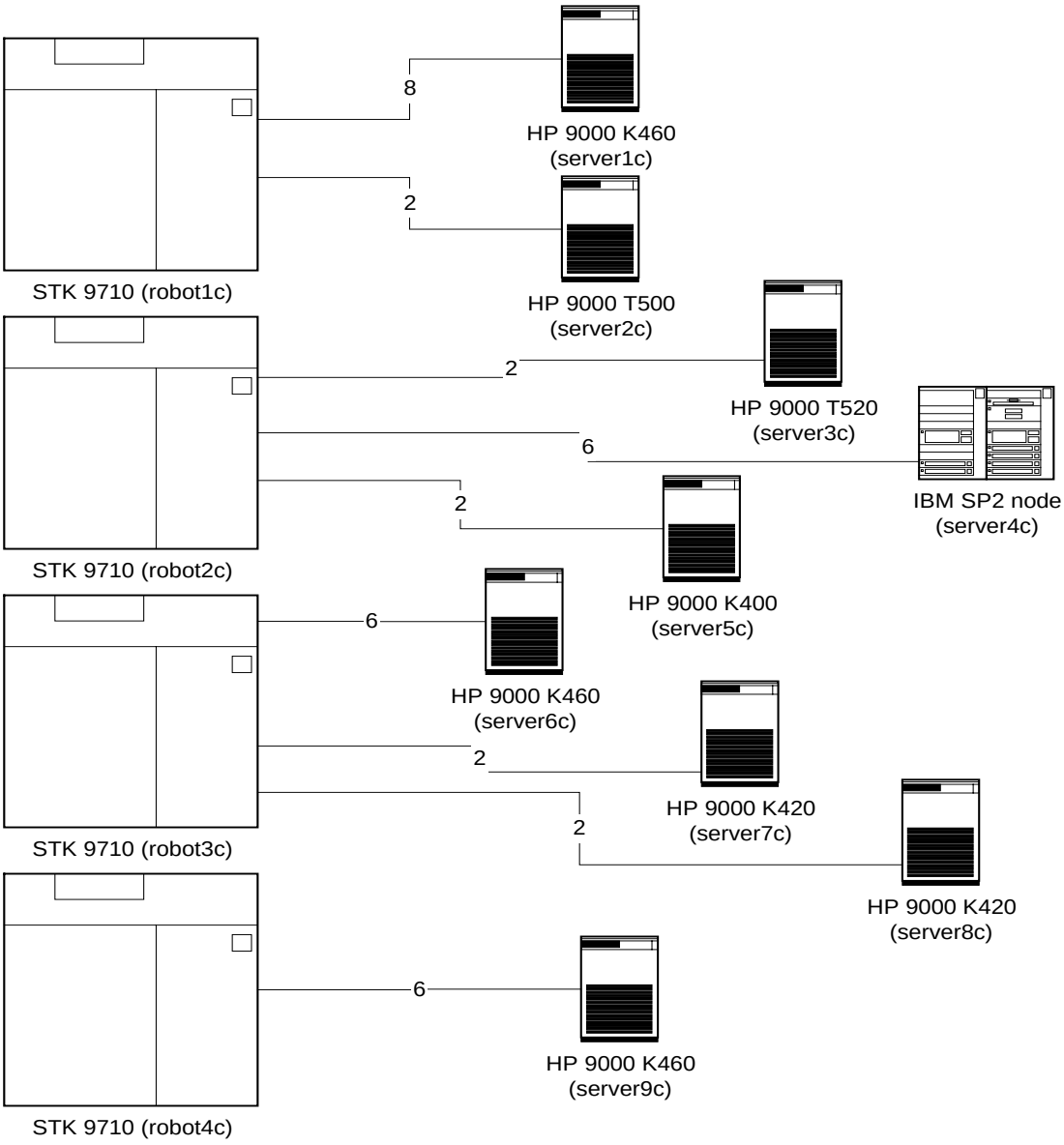


Figure 2. Example TLD Configuration

Figure 1: Example ACSLS robotic configuration

V1- Vault duplicates once per day that were written to Redwood drives

V2- Vault duplicates once per day that were written to Timberline drives

```
vault V1
bpvault_dir /usr/openv/netbackup/vault/V1
vault_type normal
drive_pairs 3
server server1b 1
server server2b 1
server server3b 1
robot_eject auto
robot_type acs
robot_num 0
acs 1
acs_lsm 0
acs_cap_0 0
acs_cap_1 1
acs_cap_count 21
acs_cap_num 2
robot_group robot1b
vault_group Offsite
starting_slot 100
pool Duplicate
dstunit server1b_RW 1
dstunit server2b_RW 1
dstunit server3b_RW 1
class os-bkup
class oracle-bkup
schedule none
duplicate_days 2
num_dbdups 1
days_holddbtape 14
dbpool NBDB-Dups
eject_header Customer B Offsite Report
vault_vendor DataBase

vault V2
bpvault_dir /usr/openv/netbackup/vault/V2
vault_type normal
drive_pairs 1
server server1b 1
robot_eject auto
robot_type acs
robot_num 0
acs 1
acs_lsm 0
acs_cap_0 0
acs_cap_1 1
acs_cap_count 21
acs_cap_num 2
robot_group robot1b-TL
vault_group Offsite-TL
starting_slot 100
pool Duplicate-TL
dstunit server1b_TL 1
class oralog-bkup
schedule none
duplicate_days 2
num_dbdups 0
dbpool NBDB-Dups
eject_header Customer B Offsite Report Timberlines
vault_vendor DataBase
```

Figure 2: Example TLD robotic configuration

V1, V2, V3, V4- Vault duplicates once per day

```
vault V1
bpvault_dir /usr/openv/netbackup/vault/V1
vault_type normal
drive_pairs 4
server server1c 3
server server2c 1
robot_eject auto
robot_type tld
robot_num 10
tld_count 14
robot_group robot1c
vault_group IronMtn-V1
pool Dups-V1
dstunit server1c-dlt 3
dstunit server2c-dlt 1
class NTservers
schedule full-wkly
duplicate_days 2
num_dbdups 0
dbpool NBDB-Dups
eject_header Customer C Bpvault Report robot1c
vault_vendor Iron Mountain

vault V2
bpvault_dir /usr/openv/netbackup/vault/V2
vault_type normal
drive_pairs 4
server server3c 1
server server4c 2
server server5c 1
robot_eject auto
robot_type tld
robot_num 11
tld_count 14
robot_group robot2c
vault_group IronMtn-V2
pool Dups-V2
dstunit server3c-dlt 1
dstunit server4c-dlt 2
dstunit server5c-dlt 1
class SP2servers
class UNIXapps
schedule none
duplicate_days 2
num_dbdups 1
days_holddbtape 7
dbpool NBDB-Dups
eject_header Customer C Bpvault Report robot2c
vault_vendor Iron Mountain
```

Figure 2: Example TLD robotic configuration (continued)

V1, V2, V3, V4- Vault duplicates once per day

```
vault V3
bpvault_dir /usr/openv/netbackup/vault/V3
vault_type normal
drive_pairs 3
server server6c 2
server server7c 1
robot_eject auto
robot_type tld
robot_num 12
tld_count 14
robot_group robot3c
vault_group IronMtn-V3
pool Dups-V3
dstunit server6c-dlt 2
dstunit server7c-dlt 1
class SAPprod
class SAPbackint
schedule none
duplicate_days 2
num_dbdups 0
dbpool NBDB-Dups
eject_header Customer C Bpvault Report robot3c
vault_vendor Iron Mountain

vault V4
bpvault_dir /usr/openv/netbackup/vault/V4
vault_type normal
drive_pairs 2
server server9c 2
robot_eject auto
robot_type tld
robot_num 13
tld_count 14
robot_group robot4c
vault_group IronMtn-V4
pool Dups-V4
dstunit server9c-dlt 2
class UNIXprod
class UNIXdata
schedule full-wkly
duplicate_days 2
num_dbdups 0
dbpool NBDB-Dups
eject_header Customer C Bpvault Report robot4c
vault_vendor Iron Mountain
```

The remaining parameter files are samples that reside in the “production” directory.

The following is the parameter file “dup_param.acs_example” for an ACSLS environment:

```
# bpvault paramater file
##
## NOTE: there should be NO EMPTY LINES in this file!!!!
#
debug 1
#   keyword: debug <0/1>
#   whether or not we want debug information
#   if enabled with a value of "1", then debug
#   information is logged to the logs found in
#   /usr/openv/netbackup/vault/logs/
#
vault V1
#   keyword: vault <name>
#   the name we've chosen for this particular
#   vault. max 5 characters.
#
bpvault_dir /usr/openv/netbackup/vault/V1
#   keyword: bpvault_dir <path>
#   the directory we've chosen for this particular
#   vault's data.
#
vault_type normal
#   keyword: vault_type <normal/original>
#   means we're duplicating instead of vaulting
#   the original tapes
#   "normal" = duplication of backup images
#   "original" = offsiting of original tapes
#
drive_pairs 2
#   keyword: drive_pairs <number>
#   how many drive PAIRS WITHIN THIS ROBOT can take
#   part in the duplication; the total number of drive
#   pairs between all server/dstunit entries
#
server server-a 1
server server-b 1
#   keyword: server <hostname> <number> <alt-hostname(s)> [ALTREADHOST] <hostname>
#   server name that controls the duplication,
#   followed by the number of drive_pairs that it
#   is controlling.
#
#   there could be multiple server lines, in the
#   event that multiple hosts backup with this
#   robot.
#
#   alt-hostname(s) are hostnames that could be listed in the
#   nbu catalog for that server, i.e. fully qualified domain
#   name. list all of them separated by spaces.
#
#   ALTREADHOST hostname used to use a different host for reading images during
#   duplication; WARNING: server and altreadhost must share the same robot
#
dstunit server-a_9840 1
dstunit server-b_9840 1
#   keyword: dstunit <storage unit name> <number>
#   the name of the netbackup storage unit
#   that will be used for this vault's
#   duplication, followed by the number
#   of drive pairs it can use for
#   duplication.
#
#   NOTE: when listing multiple servers and dstunits, they need to be listed
#   in order in this file; i.e. first server will dup to the first dstunit
#   listed, second server will dup to the second dstunit, etc.
#
robot_eject auto
#   [ see NOTEs below ]
```

```
#   keyword: robot_eject <auto/manual>
#   auto or manual, if we expect to have more
#   tapes than the cap door can hold at once,
#   manual is recommended, otherwise auto mode
#   will generate error messages and require
#   manual intervention.
#
# NOTE: the auto-generation of reports is
#       also tied to this variable.  So if
#       you want reports to be emailled or
#       printed automagically, you need to
#       set this to "auto"
#
# NOTE2: if you want to change the
#       behavior described in the NOTE:
#       above, work on the appropriate
#       bpvault.*.eject or bpvault.reports script(s)
#
robot_type acs
#   keyword: robot_type <acs/acsla/tlh/tld/tl8/tlm>
#   the type of robot we are ejecting the media from. same as robot
#   type defined in netbackup/media manager
#
robot_num 0
#   keyword: robot_num <number>
#   the robot id number we are ejecting the media from. same as robot
#   number defined in netbackup/media manager
#
acs 1
acs_lsm 0
acs_cap_0 1
acs_cap_1 2
acs_cap_count 21
acs_cap_num 2
#acs_csi_hostname acsls-server
#acs_nodrives_onmaster 0
#acs_sockname 13771
#   keyword: acs*
#   variables specific to an acs type of robot; please see sysadm
#   guide for more info.
#
robot_group 00_000_ACS
#   keyword: robot_group <vol group name>
#   volume group for volumes while
#   they are in this robot
#
vault_group Offsite
#   keyword: vault_group <vol group name>
#   volume group configured for volumes
#   while they are offsite at the vault.
#
#   NOTE: this volume group does not need to be created in
#   media manager; media will be moved to this group automatically
#   during the eject phase of bpvault.
#
#starting_slot 100
#   keyword: starting_slot <number>
#   this is an arbitrary number that will
#   be the seed slot number for your media at the offsite vendor's
#   facility.  If you have multiple vaults, make sure to avoid any duplicate
#   "slots" between vaults.
#
pool Duplicate
#   keyword: pool <pool name>
#   the media manager volume pool
#   defined for use by this robot's duplicated
#   volumes
#
#   NOTE: when vaulting originals, it is possible to list multiple
#   pools since original images could exist in different volume
#   pools; list one pool per line.
#
class os-bkup
class oracle-bkup
#class sybase-bkup
#class homedir-bkup
```

```
#   keyword: class <all/class_name>
#   keyword: classexclude <class_name>
#   you can vault explicitly by class name
#   or just specify all classes and then exclude particular classes.
#
schedule all
#schedule full
#   keyword: schedule <sched_name/all>
#   this is how we get granular, specifying
#   netbackup schedules that should be backed
#   up.
#
#   NOTE that the use of "schedule all"
#   above would actually enable duplication
#   of EVERY schedule!
#
duplicate_days 1
#   keyword: duplicate_days <number>
#   the "start" of the duplication window.
#   this is how far back bpvault
#   looks for images to duplicate.
#   this number should always be equal to
#   (or preferably larger than) the number
#   of days between vaulting sessions.
#
#   e.g. if you vault once a week, then
#   duplicate_days should be at least 7
#
#   NOTE: the higher the value of
#   duplicate_days, the longer
#   the time bpvault will take to
#   search for images before
#   it starts the duplication.
#
days_startfrom 0
#   keyword: days_startfrom <number>
#   the "end" of the duplication window,
#   i.e. number of days ago before today
#   to start searching for images to
#   duplicate.  this number should ALWAYS
#   be less than duplicate_days above.
#
#   e.g. "0" means that today's backups will
#   be eligible for duplication, whereas
#   "1" would mean that today's backups
#   would be ignored but yesterday's
#   would be eligible.
#
#dup_cleanup 1
#   keyword: dup_cleanup <0/1>
#   used to allow for
#   network duplication of other server's
#   backup images.
#   "0" = disabled
#
#   NOTE: this should be turned on if using ALTREADHOST
#   to duplicate all images no matter WHICH server was used to
#   do the backup.
#
num_dbdups 1
#   keyword: num_dbdups <number>
#   the number of duplicates of the
#   netbackup database we want to make.
#
#   NOTE: to disable bpvault's handling of
#   the netbackup db backups and
#   duplication, set this to "0"
#
#   note that only one vault
#   (the one that lives on the
#   master server) needs to have
#   this option defined.
#
dbpool NBDB_Duplicate
#   keyword: dbpool <pool name>
#   the media manager volume pool
```

```
#   configured for the offsite copies
#   of the NBU database.
#
#   NOTE: this pool cannot be
#   automatically managed via a scratch
#   pool.
#
days_holddbtape 7
#   keyword: days_holddbtape <number>
#   this is how long we want to keep
#   the duplicate DB tapes offsite.
#
#dbbackuphost server-name
#dbpath master:/usr/openv/netbackup/db
#dbpath master:/usr/openv/volmgr/database
#dbpath slave:/usr/openv/netbackup/db/media
#dbpath slave:/usr/openv/volmgr/database
#   keyword: dbbackuphost <hostname>
#   keyword: dbpath <hostname:/path>
#   used to specify the details of
#   the netbackup DB backup that
#   bpvault runs when you are NOT USING the master
#   to do the NBU DB backup. It's recommended that
#   these "dbbackuphost" and "dbpath"
#   lines are commented out, in which case
#   the values will be taken from
#   the default NetBackup configuration
#   (as seen by running the dbsyncinfo command)
#   If using another media server to do the NBU DB
#   backup, these lines MUST be defined.
#
#suspendmedia 1
#   keyword: suspendmedia <0/1>
#   turn on (1) or off (0) option to suspend original
#   media after running bpvault.
#
#   NOTE: this option is only available when vaulting
#   original media
#
#bpmedia_option suspend
#   keyword: bpmedia_option <suspend/freeze>
#   if suspending media, which option to use: both options
#   will forbid writing to the media, suspended media will
#   expire normally, frozen media will never expire.
#
#days_suspend 0
#   keyword: days_suspend <number>
#   if suspending media, the number of days from today to
#   start suspending media; days_startfrom will set the number
#   of days to from today to end.
#
#mpxdup 1
#   keyword: mpxdup <0/1>
#   setting "mpxdup" to 1 will result in
#   multiplexed duplicates - the default behavior
#   is to de-multiplex during the duplication procedure,
#   which is A Good Idea(tm) from the disaster recovery
#   standpoint, even if the de-multiplexing slows down
#   your duplication...
#
#   if your backups are multiplexed and you de-mpx during
#   duplication (set mpxdup to 0), the process will take
#   a long time. if you mpx during duplication (set mpxdup
#   to 1), then the duplication time will approach the backup
#   time.
#
#changeexpdate 1
#   keyword: changeexpdate <0/1>
#   seting this to "1" enables the ability to
#   specify modifications to the expiration
#   dates of your duplicated images.
#
#retention_length 0 7
#   keyword: retention_length <retention level> <days to add>
#   the first parameter is the retention level that will
#   be modified after duplication.  The second parameter
```

```
#   is the NUMBER OF DAYS to be added to the expiration
#   date of the original image to come up with the
#   expiration date that will be set for the duplicate
#   image.
#
#   NOTE: to specify infinite expiration for the duplicates,
#   set the days to 99999.
#
run_report 1
#   keyword: run_report <0/1>
#   whether or not we want to
#   automatically run a report after
#   a vaulting session.
#   "0" = automatic reports disabled
#   NOTE: that if you want automatic report
#   generation, you must also set robot_eject
#   to "auto", as noted above.
#
run_report_print 1
run_report_mail 1
run_report_file 1
#   keyword: run_report_print/mail/file <0/1>
#   whether or not we want to
#   generate reports to a printer, email or file.
#
report_dir /usr/openv/netbackup/vault/reports
#   keyword: report_dir <path>
#   fully qualified path to place reports
#
recoverto_days 0
#   keyword: recoverto_days <number>
#   the end of the DR window
#   used in the generation of the
#   disaster recovery report, if
#   we want a list of the tapes needed
#   to recover all of our machines
#   to as recent a state as possible.
#
recoverlength_days 7
#   keyword: recoverlength_days <number>
#   the start of the DR window
#   used in the generation of the
#   disaster recovery report, if
#   we want a list of the tapes needed
#   to recover all of our machines
#   to as recent a state as possible.
#
eject_header Customer X Offsite Report
#   keyword: eject_header <text string>
#   the header that appears on all the vaulting reports.
#
vault_vendor Arcus
#   keyword: vault_vendor <text string>
#   the vendor name that appears on all the vaulting reports.
#
#distvlt_header Distribution List for Vault
#distdtl_header Detailed Dist. List for Vault
#distsum_header Summary Dist List for Vault
#origejc_header Original Media Eject List
#origdtl_header Detailed Original Media Eject List
#picklib_header Picking List for Library
#imaglib_header Image List for Library
#pickvlt_header Picking List for Vault
#distlib_header Distribution List for Library
#invtvlt_header Inventory List for Vault
#fullvlt_header Full Inventory List for Vault
#compinv_header Complete Inventory List
#recovry_header Recovery Report for Vault
#   keyword: *_header <text string>
#   overriding header names for the various reports
#
nb_version 34
#   keyword: nb_version <netbackup version, without punctuation>
#   the version of Netbackup being run.
#
#command_log /usr/openv/netbackup/logs/bptm
```

```
#   keyword: command_log <path to specific netbackup log dir>
#   if a command_log parameter is defined,
#   then bpvault will duplicate the listed log file in its own
#   log file as well.
```

The following is the parameter file “dup_param.tld_example” for a TLD environment:

```
# bpvault paramater file
##
## NOTE: there should be NO EMPTY LINES in this file!!!!
#
debug 1
#   keyword: debug <0/1>
#   whether or not we want debug information
#   if enabled with a value of "1", then debug
#   information is logged to the logs found in
#   /usr/openv/netbackup/vault/logs/
#
vault V1
#   keyword: vault <name>
#   the name we've chosen for this particular
#   vault. max 5 characters.
#
bpvault_dir /usr/openv/netbackup/vault/V1
#   keyword: bpvault_dir <path>
#   the directory we've chosen for this particular
#   vault's data.
#
vault_type normal
#   keyword: vault_type <normal/original>
#   means we're duplicating instead of vaulting
#   the original tapes
#   "normal" = duplication of backup images
#   "original" = offsiting of original tapes
#
drive_pairs 2
#   keyword: drive_pairs <number>
#   how many drive PAIRS WITHIN THIS ROBOT can take
#   part in the duplication; the total number of drive
#   pairs between all server/dstunit entries
#
server server-a 1
server server-b 1
#   keyword: server <hostname> <number> <alt-hostname(s)> [ALTREADHOST] <hostname>
#   server name that controls the duplication,
#   followed by the number of drive_pairs that it
#   is controlling.
#
#   there could be multiple server lines, in the
#   event that multiple hosts backup with this
#   robot.
#
#   alt-hostname(s) are hostnames that could be listed in the
#   nbu catalog for that server, i.e. fully qualified domain
#   name. list all of them separated by spaces.
#
#   ALTREADHOST hostname used to use a different host for reading images during
#   duplication; WARNING: server and altreadhost must share the same robot
#
dstunit server-a_DLT 1
dstunit server-b_DLT 1
#   keyword: dstunit <storage unit name> <number>
#   the name of the netbackup storage unit
#   that will be used for this vault's
#   duplication, followed by the number
#   of drive pairs it can use for
#   duplication.
#
#   NOTE: when listing multiple servers and dstunits, they need to be listed
#   in order in this file; i.e. first server will dup to the first dstunit
#   listed, second server will dup to the second dstunit, etc.
#
robot_eject auto
#   [ see NOTEs below ]
#   keyword: robot_eject <auto/manual>
#   auto or manual, if we expect to have more
```

```
#   tapes than the cap door can hold at once,
#   manual is recommended, otherwise auto mode
#   will generate error messages and require
#   manual intervention.
#
# NOTE: the auto-generation of reports is
#       also tied to this variable.  So if
#       you want reports to be emailled or
#       printed automagically, you need to
#       set this to "auto"
#
# NOTE2: if you want to change the
#       behavior described in the NOTE:
#       above, work on the appropriate
#       bpvault.*.eject or bpvault.reports script(s)
#
robot_type tld
#   keyword: robot_type <acs/acsla/tlh/tld/tl8/tlm>
#   the type of robot we are ejecting the media from. same as robot
#   type defined in netbackup/media manager
#
robot_num 0
#   keyword: robot_num <number>
#   the robot id number we are ejecting the media from. same as robot
#   number defined in netbackup/media manager
#
tld_count 14
#   keyword: tld*
#   variables specific to a tld type of robot; please see sysadm
#   guide for more info.
#
robot_group 00_000_TLD
#   keyword: robot_group <vol group name>
#   volume group for volumes while
#   they are in this robot
#
vault_group Offsite
#   keyword: vault_group <vol group name>
#   volume group configured for volumes
#   while they are offsite at the vault.
#
#   NOTE: this volume group does not need to be created in
#   media manager; media will be moved to this group automatically
#   during the eject phase of bpvault.
#
#starting_slot 100
#   keyword: starting_slot <number>
#   this is an arbitrary number that will
#   be the seed slot number for your media at the offsite vendor's
#   facility.  If you have multiple vaults, make sure to avoid any duplicate
#   "slots" between vaults.
#
pool Duplicate
#   keyword: pool <pool name>
#   the media manager volume pool
#   defined for use by this robot's duplicated
#   volumes
#
#   NOTE: when vaulting originals, it is possible to list multiple
#   pools since original images could exist in different volume
#   pools; list one pool per line.
#
class os-bkup
class oracle-bkup
#class sybase-bkup
#class homedir-bkup
#   keyword: class <all/class_name>
#   keyword: classexclude <class_name>
#   you can vault explicitly by class name
#   or just specify all classes and then exclude particular classes.
#
schedule all
#schedule full
#   keyword: schedule <sched_name/all>
#   this is how we get granular, specifying
#   netbackup schedules that should be backed
```

```
#   up.
#
#   NOTE that the use of "schedule all"
#   above would actually enable duplication
#   of EVERY schedule!
#
duplicate_days 1
#   keyword: duplicate_days <number>
#   the "start" of the duplication window.
#   this is how far back bpvault
#   looks for images to duplicate.
#   this number should always be equal to
#   (or preferably larger than) the number
#   of days between vaulting sessions.
#
#   e.g. if you vault once a week, then
#   duplicate_days should be at least 7
#
#   NOTE: the higher the value of
#   duplicate_days, the longer
#   the time bpvault will take to
#   search for images before
#   it starts the duplication.
#
days_startfrom 0
#   keyword: days_startfrom <number>
#   the "end" of the duplication window,
#   i.e. number of days ago before today
#   to start searching for images to
#   duplicate.  this number should ALWAYS
#   be less than duplicate_days above.
#
#   e.g. "0" means that today's backups will
#   be eligible for duplication, whereas
#   "1" would mean that today's backups
#   would be ignored but yesterday's
#   would be eligible.
#
#dup_cleanup 1
#   keyword: dup_cleanup <0/1>
#   used to allow for
#   network duplication of other server's
#   backup images.
#   "0" = disabled
#
#   NOTE: this should be turned on if using ALTREADHOST
#   to duplicate all images no matter WHICH server was used to
#   do the backup.
#
num_dbdups 1
#   keyword: num_dbdups <number>
#   the number of duplicates of the
#   netbackup database we want to make.
#
#   NOTE: to disable bpvault's handling of
#   the netbackup db backups and
#   duplication, set this to "0"
#
#   note that only one vault
#   (the one that lives on the
#   master server) needs to have
#   this option defined.
#
dbpool NBDB_Duplicate
#   keyword: dbpool <pool name>
#   the media manager volume pool
#   configured for the offsite copies
#   of the NBU database.
#
#   NOTE: this pool cannot be
#   automatically managed via a scratch
#   pool.
#
days_holddbtape 7
#   keyword: days_holddbtape <number>
#   this is how long we want to keep
```

```
#   the duplicate DB tapes offsite.
#
#dbbackuphost server-name
#dbpath master:/usr/openv/netbackup/db
#dbpath master:/usr/openv/volmgr/database
#dbpath slave:/usr/openv/netbackup/db/media
#dbpath slave:/usr/openv/volmgr/database
#   keyword: dbbackuphost <hostname>
#   keyword: dbpath <hostname:/path>
#   used to specify the details of
#   the netbackup DB backup that
#   bpvault runs when you are NOT USING the master
#   to do the NBU DB backup. It's recommended that
#   these "dbbackuphost" and "dbpath"
#   lines are commented out, in which case
#   the values will be taken from
#   the default NetBackup configuration
#   (as seen by running the dbsyncinfo command)
#   If using another media server to do the NBU DB
#   backup, these lines MUST be defined.
#
#suspendmedia 1
#   keyword: suspendmedia <0/1>
#   turn on (1) or off (0) option to suspend original
#   media after running bpvault.
#
#   NOTE: this option is only available when vaulting
#   original media
#
#bpmedia_option suspend
#   keyword: bpmedia_option <suspend/freeze>
#   if suspending media, which option to use: both options
#   will forbid writing to the media, suspended media will
#   expire normally, frozen media will never expire.
#
#days_suspend 0
#   keyword: days_suspend <number>
#   if suspending media, the number of days from today to
#   start suspending media; days_startfrom will set the number
#   of days to from today to end.
#
#mpxdup 1
#   keyword: mpxdup <0/1>
#   setting "mpxdup" to 1 will result in
#   multiplexed duplicates - the default behavior
#   is to de-multiplex during the duplication procedure,
#   which is A Good Idea(tm) from the disaster recovery
#   standpoint, even if the de-multiplexing slows down
#   your duplication...
#
#   if your backups are multiplexed and you de-mpx during
#   duplication (set mpxdup to 0), the process will take
#   a long time. if you mpx during duplication (set mpxdup
#   to 1), then the duplication time will approach the backup
#   time.
#
#changeexpdate 1
#   keyword: changeexpdate <0/1>
#   seting this to "1" enables the ability to
#   specify modifications to the expiration
#   dates of your duplicated images.
#
#retention_length 0 7
#   keyword: retention_length <retention level> <days to add>
#   the first parameter is the retention level that will
#   be modified after duplication.  The second parameter
#   is the NUMBER OF DAYS to be added to the expiration
#   date of the original image to come up with the
#   expiration date that will be set for the duplicate
#   image.
#
#   NOTE: to specify infinite expiration for the duplicates,
#   set the days to 99999.
#
run_report 1
#   keyword: run_report <0/1>
```

```
#   whether or not we want to
#   automatically run a report after
#   a vaulting session.
#   "0" = automatic reports disabled
#   NOTE: that if you want automatic report
#   generation, you must also set robot_eject
#   to "auto", as noted above.
#
run_report_print 1
run_report_mail 1
run_report_file 1
#   keyword: run_report_print/mail/file <0/1>
#   whether or not we want to
#   generate reports to a printer, email or file.
#
report_dir /usr/openv/netbackup/vault/reports
#   keyword: report_dir <path>
#   fully qualified path to place reports
#
recoverto_days 0
#   keyword: recoverto_days <number>
#   the end of the DR window
#   used in the generation of the
#   disaster recovery report, if
#   we want a list of the tapes needed
#   to recover all of our machines
#   to as recent a state as possible.
#
recoverlength_days 7
#   keyword: recoverlength_days <number>
#   the start of the DR window
#   used in the generation of the
#   disaster recovery report, if
#   we want a list of the tapes needed
#   to recover all of our machines
#   to as recent a state as possible.
#
eject_header Customer X Offsite Report
#   keyword: eject_header <text string>
#   the header that appears on all the vaulting reports.
#
vault_vendor Arcus
#   keyword: vault_vendor <text string>
#   the vendor name that appears on all the vaulting reports.
#
#distvlt_header Distribution List for Vault
#distdtl_header Detailed Dist. List for Vault
#distsum_header Summary Dist List for Vault
#origejc_header Original Media Eject List
#origdtl_header Detailed Original Media Eject List
#picklib_header Picking List for Library
#imaglib_header Image List for Library
#pickvlt_header Picking List for Vault
#distlib_header Distribution List for Library
#invtvlt_header Inventory List for Vault
#fullvlt_header Full Inventory List for Vault
#compinv_header Complete Inventory List
#recovry_header Recovery Report for Vault
#   keyword: *_header <text string>
#   overriding header names for the various reports
#
nb_version 34
#   keyword: nb_version <netbackup version, without punctuation>
#   the version of Netbackup being run.
#
#command_log /usr/openv/netbackup/logs/bptm
#   keyword: command_log <path to specific netbackup log dir>
#   if a command_log parameter is defined,
#   then bpvault will duplicate the listed log file in its own
#   log file as well.
```

Appendix D – Output File: Preview Mode

```
********************************************************************************
Program:               bpvault
Profile:                      /usr/openv/netbackup/vault/CH_V1/DUP47/dup_param_wfb
Vault:                 CH_V1
Duplicate:             DUP47
Function:              PREVIEW
Preview File:          /usr/openv/netbackup/vault/CH_V1/DUP47/dup.preview.out
Log File:              /usr/openv/netbackup/vault/CH_V1/log.file
Error File:            /usr/openv/netbackup/vault/CH_V1/DUP47/bpvault.error.file
********************************************************************************
bpduplicate: 07:09:57 INF - Skipping backup id fapollo_0862840895, it already has 2 copies.
bpduplicate:
bpduplicate: 07:10:02 INF - Skipping backup id fapollo_0862754553, it already has 2 copies.
bpduplicate:
bpduplicate: bpvault: completed bpduplicate preview mode for DUP47
bpduplicate: 07:10:30 INF - Skipping backup id fbentley_0862904449, it already has 2 copies.
bpduplicate:
bpduplicate: 07:10:36 INF - Skipping backup id fbentley_0862818074, it already has 2 copies.
bpduplicate:
bpduplicate: 07:10:45 INF - Skipping backup id fbentley_0862797764, it already has 2 copies.
bpduplicate:
bpduplicate: bpvault: completed bpduplicate preview mode for DUP47
bpduplicate: 07:11:13 INF - found no images or media matching the selection criteria
bpduplicate: bpvault: completed bpduplicate preview mode for DUP47
bpduplicate: 07:11:40 INF - found no images or media matching the selection criteria
bpduplicate: bpvault: completed bpduplicate preview mode for DUP47
bpvault: do_preview: there are 25 images to duplicate for DUP47
********************************************************************************
```

Appendix E – Output File: Duplication Mode

This Appendix shows bpvault output file for duplicate. It shows how 1 master and 2 slave servers can run duplications simultaneously:

```
*******************************************************************************
Program:              bpvault
Profile:                    /usr/openv/netbackup/vault/CH_V1/DUP47/dup_param_wfb
Vault:                CH_V1
Duplicate:            DUP47
Function:             DUPLICATE
Preview File:         /usr/openv/netbackup/vault/CH_V1/DUP47/dup.preview.out
Log File:             /usr/openv/netbackup/vault/CH_V1/log.file
Error File:           /usr/openv/netbackup/vault/CH_V1/DUP47/bpvault.error.0
*******************************************************************************
bpvault: copied 12 image(s) for server nirvana, # of drive_pair(s) 2
bpvault: copied 2 image(s) for server inxs, # of drive_pair(s) 1
bpvault: copied 11 image(s) for server zeppelin, # of drive_pair(s) 2
bpvault[0]: nirvana[0]: starting bpduplicate[0]: nirvana[0] for image fapollo_0862927360 at 05/07/97
07:27:01
bpduplicate[0]: nirvana[0]: Duplicate started Wed May 7 07:27:01 1997
bpduplicate[0]: nirvana[0]:
bpduplicate[0]: nirvana[0]: 07:27:06 INF - Destination storage unit b_nirvana on host nirvana.
bpduplicate[0]: nirvana[0]: 07:27:09 INF - Duplicating class apollo_BR1, schedule Incr_Daily
(fapollo_0862927360) created 05/06/97 07:02:40, media id S05861.
bpvault[1]: nirvana[1]: starting bpduplicate[1]: nirvana[1] for image fbentley_0862990597 at 05/07/97
07:27:01
bpduplicate[1]: nirvana[1]: Duplicate started Wed May 7 07:27:02 1997
bpduplicate[1]: nirvana[1]:
bpduplicate[1]: nirvana[1]: 07:27:05 INF - Destination storage unit d_nirvana on host nirvana.
bpduplicate[1]: nirvana[1]: 07:27:08 INF - Duplicating class bentley_BR1, schedule daily-incr
(fbentley_0862990597) created 05/07/97 00:36:37, media id S05367.
bpvault[2]: inxs[0]: starting bpduplicate[2]: inxs[0] for image fcapt01_0862970576 at 05/07/97
07:27:01
bpduplicate[2]: inxs[0]: Duplicate started Wed May 7 07:27:02 1997
bpduplicate[2]: inxs[0]:
bpduplicate[2]: inxs[0]: 07:27:04 INF - Destination storage unit e_inxs on host inxs.
bpduplicate[2]: inxs[0]: 07:27:07 INF - Duplicating class capt01_BR1, schedule daily-incr
(fcapt01_0862970576) created 05/06/97 19:02:56, media id S08498.
bpvault[3]: zeppelin[0]: starting bpduplicate[3]: zeppelin[0] for image femerald_0862966915 at
05/07/97 07:27:01
bpduplicate[3]: zeppelin[0]: Duplicate started Wed May 7 07:27:02 1997
bpduplicate[3]: zeppelin[0]:
bpduplicate[3]: zeppelin[0]: 07:27:04 INF - Destination storage unit a_zeppelin on host zeppelin.
bpduplicate[3]: zeppelin[0]: 07:27:07 INF - Duplicating class emerald_BR1, schedule Incr_Daily
(femerald_0862966915) created 05/06/97 18:01:55, media id S05765.
bpduplicate[3]: zeppelin[0]: 07:27:16 INF - Waiting for mount of media id S00002 on server zeppelin.
bpvault[4]: zeppelin[1]: starting bpduplicate[4]: zeppelin[1] for image fgolddust_0862977806 at
05/07/97 07:27:01
bpduplicate[4]: zeppelin[1]: Duplicate started Wed May 7 07:27:02 1997
bpduplicate[4]: zeppelin[1]:
bpduplicate[4]: zeppelin[1]: 07:27:04 INF - Destination storage unit c_zeppelin on host zeppelin.
bpduplicate[4]: zeppelin[1]: 07:27:07 INF - Duplicating class golddust_BR1, schedule Incr_Daily
(fgolddust_0862977806) created 05/06/97 21:03:26, media id S05994.
bpduplicate[2]: inxs[0]: 07:27:17 INF - Waiting for mount of media id S00814 on server inxs.
bpduplicate[1]: nirvana[1]: 07:27:25 INF - Waiting for mount of media id S03213 on server nirvana.
bpduplicate[2]: inxs[0]: 07:27:33 INF - Waiting for mount of media id S08498 on server inxs.
bpduplicate[4]: zeppelin[1]: 07:27:37 INF - Waiting for mount of media id S00453 on server zeppelin.
bpduplicate[0]: nirvana[0]: 07:27:37 INF - Waiting for mount of media id S00391 on server nirvana.
bpduplicate[1]: nirvana[1]: 07:27:43 INF - Waiting for mount of media id S05367 on server nirvana.
bpduplicate[2]: inxs[0]: 07:27:50 INF - Beginning duplication on server inxs of client fcapt01 image,
creating copy 2.
bpduplicate[0]: nirvana[0]: 07:27:52 INF - Waiting for mount of media id S05861 on server nirvana.
bpduplicate[3]: zeppelin[0]: 07:27:52 INF - Beginning duplication on server zeppelin of client
femerald image, creating copy 2.
bpduplicate[4]: zeppelin[1]: 07:27:52 INF - Waiting for mount of media id S05994 on server zeppelin.
bpduplicate[1]: nirvana[1]: 07:27:55 INF - Waiting for positioning of media id S03213 on server
nirvana.
bpduplicate[3]: zeppelin[0]: 07:27:56 INF - Waiting for mount of media id S05765 on server zeppelin.
```

```
bpduplicate[4]: zeppelin[1]: 07:28:08 INF - Beginning duplication on server zeppelin of client
fgolddust image, creating copy 2.
bpduplicate[2]: inxs[0]: 07:28:09 INF - Waiting for positioning of media id S08498 on server inxs.
bpduplicate[1]: nirvana[1]: 07:28:11 INF - Waiting for positioning of media id S05367 on server
nirvana.
bpduplicate[0]: nirvana[0]: 07:28:12 INF - Waiting for positioning of media id S00391 on server
nirvana.
bpduplicate[3]: zeppelin[0]: 07:28:14 INF - Waiting for positioning of media id S05765 on server
zeppelin.
bpduplicate[1]: nirvana[1]: 07:28:15 INF - Beginning duplication on server nirvana of client fbentley
image, creating copy 2.
bpduplicate[2]: inxs[0]: 07:28:28 INF - Beginning duplicate on server inxs of client fcapt01.
bpduplicate[0]: nirvana[0]: 07:28:30 INF - Waiting for positioning of media id S05861 on server
nirvana.
bpduplicate[1]: nirvana[1]: 07:28:34 INF - Beginning duplicate on server nirvana of client fbentley.
bpduplicate[1]: nirvana[1]: 07:28:41 INF - Reading True Image Recovery information from media id
S05367.
bpduplicate[1]: nirvana[1]: 07:28:45 INF - Begin writing True Image Recovery information for copy 2.
bpduplicate[4]: zeppelin[1]: 07:28:45 INF - Waiting for positioning of media id S05994 on server
zeppelin.
bpduplicate[1]: nirvana[1]: 07:28:55 INF - Duplicate of backupid fbentley_0862990597 successful.
bpduplicate[1]: nirvana[1]:
bpduplicate[1]: nirvana[1]: 07:28:58 INF - Status = successfully duplicated 1 of 1 images.
bpvault[1]: nirvana[1]: finished bpduplicate[1]: nirvana[1] for image fbentley_0862990597 at 05/07/97
07:28:58
```

…deleted remaining log entries…..

****

Appendix F – Output File: Duplication Mode Debugging

This Appendix shows part of and example output file with debugging turned on, and with “bptm” log also monitored. You can see how bptm and bpduplicate progress is interleaved by mounts etc.. This was done for 2 drive pairs, but one pair was duplicating to disk:

```
init: debug: 1
init: drive_pairs: 2
init: server[0].name: horse, server[0].drive_pair: 2, server[0].offset: 0,
server_pair_total: 2
init: dstunit[0]: tape4mm
init: dstunit[1]: testdump
init: duplicate_days: 45
init: vault: V1
init: vault_vendor: BLAP
init: class[0]: DUPTHIS
init: robot_group: robot_grp
init: vault_group: vault_grp
init: copy_number: 1
init: schedule: none
init: pool: nb_duplicates
init: bpvault_dir: /usr/openv/netbackup/vault/V1
init: dup_file template: bpvault.dup.
init: error_file template: bpvault.error.
init: log_file template: bpvault.log.
init: eject_header: Wells Fargo - Cassie Hills
init: duplicate_command: /usr/openv/netbackup/bin/admincmd/bpduplicate -v
init: robot_inventory: /usr/openv/netbackup/vault/bin/robot_inventory
init: eject_command: /usr/openv/netbackup/vault/bin/eject_tapes
init: query_command: /usr/openv/volmgr/bin/vmquery
init: change_command: /usr/openv/volmgr/bin/vmchange
command_log: /usr/openv/netbackup/logs/bptm
init: debug_counter: 20
init: dup_files[0] = bpvault.dup.0
init: log_files[0] = bpvault.log.0
init: dup_files[1] = bpvault.dup.1
init: log_files[1] = bpvault.log.1
init: start date for duplication = 03/23/97, end date = 05/07/97
*******************************************************************************
Program:              bpvault
Profile:              /usr/openv/netbackup/vault/V1/DUP132/dup_param
Vault:                V1
Duplicate:            DUP132
Function:             DUPLICATE
Preview File:         /usr/openv/netbackup/vault/V1/DUP132/dup.preview.out
Log File:             /usr/openv/netbackup/vault/log.file
Error File:    /usr/openv/netbackup/vault/V1/DUP132/bpvault.error.0
*******************************************************************************
bpvault: entering do_dup_ms
bpvault: entering load_image
bpvault: do_dup_ms: opening dup_files[0] for server: horse[0], drive_pair: 0: DUP132/bpvault.dup.0
bpvault: do_dup_ms: opening dup_files[1] for server: horse[0], drive_pair: 1: DUP132/bpvault.dup.1
bpvault: entering save_image
bpvault: save_image: found image to duplicate on server[0]: horse[0]: 05/01/97 23:32:05 DUPTHIS
immediate horse.ranch.dpoints.com_0862554725 horse RNCH01
bpvault: entering save_image
bpvault: save_image: found image to duplicate on server[0]: horse[1]: 05/03/97 00:06:14 DUPTHIS
immediate horse.ranch.dpoints.com_0862643174 horse RNCH01
bpvault: entering save_image
bpvault: save_image: found image to duplicate on server[0]: horse[0]: 05/04/97 01:57:31 DUPTHIS
immediate horse.ranch.dpoints.com_0862736251 horse RNCH01
bpvault: entering save_image
bpvault: save_image: found image to duplicate on server[0]: horse[1]: 05/05/97 12:03:16 DUPTHIS
immediate horse.ranch.dpoints.com_0862858996 horse RNCH01
bpvault: entering save_image
bpvault: save_image: found image to duplicate on server[0]: horse[0]: 05/06/97 00:08:30 DUPTHIS
immediate horse.ranch.dpoints.com_0862902510 horse RNCH01
bpvault: entering save_image
```

```
bpvault: save_image: found image to duplicate on server[0]: horse[1]: 04/14/97 12:23:15 DUPTHIS
immediate horse.ranch.dpoints.com_0861045795 horse RNCH05
bpvault: copied 6 image(s) for server horse, # of drive_pair(s) 2
bpvault: entering log_monitors
bpvault: log_monitors: monitoring log , log file: /usr/openv/netbackup/logs/bptm/log.050797
bpvault: entering child_monitor for
bpvault: child_monitor: in directory /, subdir , open log to monitor:
/usr/openv/netbackup/logs/bptm/log.050797
bpvault: log_monitors: forked log monitor process: 28634
bpvault: calling dup_normal
bpvault: entering dup_normal
bpvault[0]: horse[0]: dup_normal: starting duplicates on bpduplicate[0]: horse[0]: 0
bpvault[0]: horse[0]: entering load_image
bpvault[0]: horse[0]: dup_normal: found image to duplicate: 05/01/97 23:32:05 DUPTHIS immediate
horse.ranch.dpoints.com_0862554725 horse s
bpvault[0]: horse[0]: dup_normal: running command: cd /usr/openv/netbackup/vault/V1; cd DUP132; rm
-f bpvault.error.0; touch bpvault.error.0; /usr/openv/netbackup/bin/admincmd/bpduplicate -v -
dstunit tape4mm -backupid horse.ranch.dpoints.com_0862554725 -L
/usr/openv/netbackup/vault/V1/DUP132/bpvault.error.0 -dp nb_duplicates -cn 1 >> bpvault.log.0 2>&1
bpvault: dup_normal: forked dup process: 28638
bpvault[1]: horse[1]: dup_normal: starting duplicates on bpduplicate[1]: horse[1]: 1
bpvault[1]: horse[1]: entering load_image
bpvault[1]: horse[1]: dup_normal: found image to duplicate: 05/03/97 00:06:14 DUPTHIS immediate
horse.ranch.dpoints.com_0862643174 horse s
bpvault[1]: horse[1]: dup_normal: running command: cd /usr/openv/netbackup/vault/V1; cd DUP132; rm
-f bpvault.error.1; touch bpvault.error.1; /usr/openv/netbackup/bin/admincmd/bpduplicate -v -
dstunit testdump -backupid horse.ranch.dpoints.com_0862643174 -L
/usr/openv/netbackup/vault/V1/DUP132/bpvault.error.1 -dp nb_duplicates -cn 1 >> bpvault.log.1 2>&1
bpvault: dup_normal: forked dup process: 28641
bpvault: entering wait_dup
15:55:10 [28670] <2> INITIATING: -dup -cmd -nosig -v -L
/usr/openv/netbackup/vault/V1/DUP132/bpvault.error.0 -ru root -rclnt horse -c
horse.ranch.dpoints.com -b horse.ranch.dpoints.com_0862554725 -cl DUPTHIS -sl immediate -bt
862554725 -st 0 -rl 3 -ct 0 -cj 1 -den 12 -rt 0 -p nb_duplicates -date 865233125 -cn 2
15:55:10 [28670] <2> io_set_recvbuf: setting receive network buffer to 32032 bytes
15:55:10 [28670] <2> io_init: using 32768 data buffer size
15:55:10 [28670] <2> io_init: CINDEX 0, sched Kbytes for monitoring = 10000
15:55:10 [28670] <2> io_init: using 8 data buffers
15:55:10 [28670] <2> io_init: child delay = 20, parent delay = 30 (milliseconds)
15:55:10 [28670] <2> io_init: shm_size = 262244, buffer address = 0xef5d0000, buf control =
0xef610000, ready ptr = 0xef610060
15:55:11 [28670] <2> select_media: skipping media id TEST01, it is not the correct retention_level
15:55:11 [28670] <2> select_media: skipping media id RNCH05, either FULL or FROZE or SUSPENDED or
IMPORTED
15:55:11 [28670] <2> select_media: skipping media id RNCH01, it is not the correct density
15:55:11 [28670] <2> select_media: skipping media id RNCH02, it is not the correct density
15:55:11 [28670] <2> standalone_select_media: found RVSN DUP005 in device 1
15:55:11 [28670] <2> vmdb_query_byID_getpool: server returned: 1 DUP005 9 ------------- 0 0 NONE
robot_grp 0 0 0 0 0 main root root 0 1 855483286 0 862510530 0 0 8 21 0 0 0 855483336 - None

15:55:11 [28670] <2> vmdb_query_byID_getpool: server returned: 1 16 NetBackup ANYHOST 0 -2 the
NetBackup pool

15:55:11 [28670] <4> standalone_select_media: skipping mounted media id DUP005, it is not
NOT_ROBOTIC, not correct media type, or not in correct volume pool
15:55:11 [28670] <2> pooldb_query: server returned: 0 16 None ANYHOST -1 -2 the None pool (for
anyone)

15:55:11 [28670] <2> pooldb_query: server returned: 1 16 NetBackup ANYHOST 0 -2 the NetBackup pool

15:55:11 [28670] <2> pooldb_query: server returned: 3 21 nb_duplicates ANYHOST -1 -2 Duplication
Pool

15:55:11 [28670] <2> pooldb_query: server returned: 2 21 Duplicates ANYHOST -1 -2 Wells Fargo
Cassie Hill

15:55:11 [28670] <2> pooldb_query: server returned: 4 21 testpool ANYHOST -1 -2 test for vmd

15:55:11 [28670] <2> pooldb_query: server returned: REQUEST COMPLETE
15:55:11 [28670] <2> standalone_select_media: pool ordinal for nb_duplicates is 3
15:55:11 [28670] <2> select_media: cannot find mounted media in standalone drive, select media
from media database if possible
15:55:11 [28670] <2> select_media: skipping media id TEST01, it is not the correct retention_level
15:55:11 [28670] <2> select_media: skipping media id RNCH05, either FULL or FROZE or SUSPENDED or
IMPORTED
15:55:11 [28670] <2> select_media: skipping media id RNCH01, it is not the correct density
```

```
15:55:11 [28670] <2> select_media: skipping media id RNCH02, it is not the correct density
15:55:11 [28670] <2> select_media: getting new media id for retention level 3
15:55:12 [28676] <2> INITIATING: -copy -cmd -nosig -everything -cn 1 -c horse.ranch.dpoints.com -b
horse.ranch.dpoints.com_0862643174 -port 1017 -1 horse -L
/usr/openv/netbackup/vault/V1/DUP132/bpvault.error.1 -ru root -rclnt horse -v
15:55:12 [28676] <2> connect_data_socket: data socket connected is 6, duping to stdout
15:55:12 [28670] <2> vmdb_query_scratch_bypool2: server returned: 1 DUP003 9 ------------- 0 0
NONE robot_grp 0 0 0 0 0 main root root 0 3 855190099 863045710 856035725 0 0 7 21 0 0 0 855742053
- V1|DUP101|S2|00000000

15:55:12 [28670] <2> db_byid: search for media id DUP003
15:55:12 [28670] <2> db_put: write media id DUP003, offset = 4
15:55:12 [28670] <2> select_media: selected media id DUP003 for backup, horse.ranch.dpoints.com(rl
= 3) <----------
15:55:12 [28670] <2> create_data_socket: writing port numbers 1010 -1 to stderr
15:55:13 [28676] <2> read_backup: media id RNCH01, copy 1, fragment 1 (24928 Kbytes) being
considered for duplicate
15:55:13 [28676] <2> db_byid: search for media id RNCH01
15:55:13 [28676] <2> db_byid: RNCH01 found at offset 2
15:55:13 [28676] <2> vmdb_query_byID_getpool: server returned: 1 RNCH01 4 ------------- 0 -1 NONE
--- 0 0 0 0 0 main root root 0 1 841770512 862542085 862902515 0 0 29 21 0 0 0 841770567 - Ranch

15:55:13 [28676] <2> vmdb_query_byID_getpool: server returned: 1 16 NetBackup ANYHOST 0 -2 the
NetBackup pool

15:55:13 [28676] <2> mount_open_media: Waiting for mount of media id RNCH01 on server horse.
15:55:14 [28676] <2> mount_open_media: setting mount timeout to 72000 seconds
15:55:14 [28683] <2> INITIATING: -copy -cmd -nosig -everything -cn 1 -c horse.ranch.dpoints.com -b
horse.ranch.dpoints.com_0862554725 -port 1010 -1 horse -L
/usr/openv/netbackup/vault/V1/DUP132/bpvault.error.0 -ru root -rclnt horse -v
15:55:15 [28670] <2> create_data_socket: data socket created is 7, duping to stdin
15:55:15 [28676] <2> io_open: file /usr/openv/netbackup/db/media/tpreq/RNCH01 successfully opened
15:55:15 [28676] <2> io_read_media_header: drive index 0, reading media header, buflen = 32768,
buff = 0xa24f8
15:55:15 [28676] <2> io_ioctl: command (5)MTREW 1 from (bptm.c.3302) on drive index 0
15:55:15 [28670] <2> write_backup: backup child process is pid 28685
15:55:15 [28670] <2> mount_open_media: Waiting for mount of media id DUP003 on server horse.
15:55:15 [28683] <2> connect_data_socket: data socket connected is 6, duping to stdout
15:55:15 [28683] <2> read_backup: media id RNCH01, copy 1, fragment 1 (24608 Kbytes) being
considered for duplicate
```

...rest of log deleted....

```
bpvault: wait_dup: monitor program PID 28638 returned
bpvault: wait_dup: monitor program PID 28634 returned
bpvault: wait_dup: command log process returned: 28634
bpvault: entering kill_monitors
bpvault: kill_monitors: send signal 9 to backint monitor PID 28641
bpvault: do_dup_ms: start time: 05/07/97 15:54 end time 05/07/97 16:43
********************************************************************************
```

Appendix G – Log File

Sample log file showing “error” and “ok” duplication status on per image basis, without all the other information shown in the output file. This file log can be easily used to link with Event Management software.

```
bpvault[1]: horse[1]: status: error, image horse.ranch.dpoints.com_0861045795, start: 05/07/97
15:51:37, end: 05/07/97 15:52:01
bpvault[0]: horse[0]: found ERROR: vault: V1, session: DUP132, error line: 15:52:02 INF - The
operator on server horse denied the mount request.

bpvault[0]: horse[0]: status: error, image horse.ranch.dpoints.com_0862736251, start: 05/07/97
15:51:30, end: 05/07/97 15:52:02
bpvault[0]: horse[0]: found ERROR: vault: V1, session: DUP132, error line: 15:52:31 INF - The
operator on server horse denied the mount request.

bpvault[0]: horse[0]: status: error, image horse.ranch.dpoints.com_0862902510, start: 05/07/97
15:52:10, end: 05/07/97 15:52:32
bpvault[1]: horse[1]: status: ok, image horse.ranch.dpoints.com_0862643174, start: 05/07/97
15:55:03, end: 05/07/97 15:57:28
bpvault[0]: horse[0]: status: ok, image horse.ranch.dpoints.com_0862554725, start: 05/07/97
15:55:03, end: 05/07/97 15:59:24
bpvault[1]: horse[1]: status: ok, image horse.ranch.dpoints.com_0862858996, start: 05/07/97
15:57:32, end: 05/07/97 16:00:48
bpvault[0]: horse[0]: status: ok, image horse.ranch.dpoints.com_0862736251, start: 05/07/97
15:59:25, end: 05/07/97 16:02:08
bpvault[0]: horse[0]: status: ok, image horse.ranch.dpoints.com_0862902510, start: 05/07/97
16:02:09, end: 05/07/97 16:05:56
```

Appendix H – Log File (Optimized Duplication)

This log file is similar to the previous Appendix except it was generated from the optimized “Duplication by Media” mode (see “Procedures – Duplication” for more details). Notice that as each image is duplicated within the batch a message is displayed. When the entire batch is completed, a final message is displayed with either an “error”, “ok” or “partially successful” status.

```
./bpvault: started 1 bpduplicate servers
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0901571723, batch count 1 of 3,
vault: V1, session: DUP56
./bpvault[0]: dogbert[0]: found IMAGE ERROR: vault: V1, session: DUP56, error line: 22:34:25 INF
- host dogbert backup id dogbert_0901571933 read failed, media manager killed by signal (82).

./bpvault[0]: dogbert[0]: found IMAGE ERROR: vault: V1, session: DUP56, error line: 22:35:28 INF
- host dogbert backupid dogbert_0901571933 write failed, media manager killed by signal (82).

./bpvault[0]: dogbert[0]: failed batch dup of image dogbert_0901571933, batch count 2 of 3,
vault: V1, session: DUP56, error line: 22:35:30 INF - Duplicate of backup id dogbert_0901571933
failed, media manager killed by signal (82).

./bpvault[0]: dogbert[0]: successful dup of image dogbert_0901572406, count 3 of 3, vault: V1,
session: DUP56
./bpvault[0]: dogbert[0]: found PARTIALLY SUCCESSFUL: vault: V1, session: DUP56
./bpvault[0]: dogbert[0]: status: partially successful, 3 images on media A00003, start:
07/27/98 22:28:49, end: 07/27/98 22:38:27, batch 1 of 2

...

./bpvault: started 1 bpduplicate servers
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903114907, batch count 1 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903115109, batch count 2 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903116164, batch count 3 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: status: ok, 3 images on media A00003, start: 08/17/98 22:39:21, end:
08/17/98 22:48:18, batch 1 of 2
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903116733, batch count 1 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903117334, batch count 2 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: successful batch dup of image dogbert_0903119308, batch count 3 of 3,
vault: V1, session: DUP68
./bpvault[0]: dogbert[0]: status: ok, 3 images on media A00004, start: 08/17/98 22:48:21, end:
08/17/98 22:58:09, batch 2 of 2
```